(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 734 504**

(51) Int. CI.:

G06F 17/30 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **15.03.2013 PCT/EP2013/055455**

(87) Fecha y número de publicación internacional: **19.09.2013 WO13135887**

(96) Fecha de presentación y número de la solicitud europea: **15.03.2013 E 13714565 (2)**

(97) Fecha y número de publicación de la concesión europea: **01.05.2019 EP 2825982**

(54) Título: **Método para proporcionar una estructura de índice en una base de datos**

(30) Prioridad:

16.03.2012 US 201261611865 P

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
10.12.2019

(73) Titular/es:

CAPISH INTERNATIONAL AB (100.0%)
Carlsgatan 3
211 20 Malmö , SE

(72) Inventor/es:

DREWS, CARL HENRIK y
GESTRELIUS, STAFFAN

(74) Agente/Representante:

TOMAS GIL, Tesifonte Enrique

Aviso: En el plazo de nueve meses a contar desde la fecha de publicación en el Boletín Europeo de Patentes, de la mención de concesión de la patente europea, cualquier persona podrá oponerse ante la Oficina Europea de Patentes a la patente concedida. La oposición deberá formularse por escrito y estar motivada; sólo se considerará como formulada una vez que se haya realizado el pago de la tasa de oposición (art. 99.1 del Convenio sobre Concesión de Patentes Europeas).

DESCRIPCIÓN

Método para proporcionar una estructura de índice en una base de datos

CAMPO TÉCNICO

[0001] La mayor parte de la información creada por las organizaciones de producción de conocimiento se captura mediante diferentes sistemas de información que se optimizan para una parte específica de la cadena de valor. Mediante el uso de varias técnicas de integración, la información de los diferentes sistemas de información se puede integrar en conjuntos de datos combinados que se pueden consultar y analizar de forma adicional para desbloquear el conocimiento completo producido por la organización. Dado que los conjuntos de datos se combinan desde múltiples fuentes y se pueden actualizar con frecuencia, el sistema de información que manipula los conjuntos de datos combinados debe ser capaz de manejar tanto actualizaciones frecuentes como cambios estructurales en los conjuntos de datos de una forma flexible y sencilla. Asimismo, con el objetivo de permitir que el conocimiento de los conjuntos de datos combinados se desbloquee, el sistema de información debe permitir que los conjuntos de datos combinados se consulten y analicen sin imponer restricciones técnicas. El reto de acceder a los conjuntos de datos de esta manera es una realidad en la mayor parte de organizaciones de producción de conocimiento que buscan maximizar el conocimiento adquirido a partir de los datos producidos.

ESTADO DE LA TÉCNICA

[0002] Las soluciones actuales que se encargan de estos problemas pueden, desde una perspectiva de usuario, dividirse en tres categorías principales. La primera categoría de soluciones proporciona al usuario un acceso completo al modelo de datos subyacente y, por tanto, permite que los conjuntos de datos se consulten y analicen de manera flexible. Sin embargo, dado que el usuario debe tener un buen conocimiento del modelo de datos subyacente y el lenguaje de consulta utilizado con frecuencia es complejo, para permitir consultas complejas, por lo general, esta categoría de soluciones se considera difícil de usar para un usuario medio. Del mismo modo, dado que las consultas y los análisis efectuados en los conjuntos de datos se llevan a cabo directamente en el modelo de datos, las soluciones de esta categoría son sensibles a los cambios en la estructura de los datos.

[0003] La segunda categoría de soluciones comprende el modelo de datos subyacente del usuario y, en cambio, usa un modelo de información personalizado para un dominio de problema específico y utiliza reglas predefinidas para traducir entre el modelo de información y el modelo de datos. Dado que esta categoría de soluciones se sirve de una interfaz personalizada para consultar y analizar un dominio de problema específico, los conjuntos de datos se pueden consultar y analizar de manera sencilla. Sin embargo, dado que la solución se crea para un dominio de problema específico, la mayoría de los sistemas de esta categoría no permite que los conjuntos de datos se consulten y analicen de manera flexible, sino que los tipos de consultas y análisis posibles se ven limitados por el diseño del sistema de información. Además, las soluciones de esta categoría son muy sensibles a los cambios en la estructura de los datos dado que el sistema se ha creado encima de los conjuntos de datos predefinidos.

[0004] La tercera categoría de soluciones da acceso al usuario al modelo de datos subyacente, pero intenta llevar a cabo muchas tareas complejas (por ejemplo conectar las partes relacionadas del conjunto de datos) automáticamente. Esto hace que sea posible consultar y analizar los conjuntos de datos de manera flexible y sencilla, ya que el usuario puede acceder al modelo de datos directamente, pero no necesita conocer el modelo de datos completo o un lenguaje de consulta complejo para explorar los conjuntos de datos. Sin embargo, con las soluciones de esta categoría, con frecuencia es muy complejo eludir el comportamiento automático y crear un patrón de consulta y análisis diferente del comportamiento automático. Por lo tanto, la flexibilidad disponible durante la consulta y el análisis de los conjuntos de datos se ve limitada por el diseño de la solución. Algunas soluciones de esta categoría, dependiendo de la implementación técnica, también son sensibles a los cambios en la estructura de los conjuntos de datos.

[0005] Todas las categorías de soluciones anteriormente descritas imponen restricciones a la flexibilidad o la sencillez de la consulta y el análisis de los conjuntos de datos. La mayor parte de las soluciones disponibles se ven perjudicadas por ser sensibles a los cambios estructurales de los conjuntos de datos. Por lo tanto, existe una clara necesidad de mejorar la situación sugiriendo un método y una tecnología nuevos para consultar y analizar los conjuntos de datos de manera flexible y sencilla sin imponer restricciones y mientras se permite todavía que los conjuntos de datos cambien estructuralmente.

[0006] La patente WO 00/72165 A1 divulga un mercado de datos que usa vectores de bits para generar vistas de datos.

RESUMEN DE LA INVENCIÓN

[0007] Un objetivo de la invención es encargarse de los problemas mencionados anteriormente y mitigarlos, así como proporcionar un nuevo método con tecnología de soporte para hacer que el conocimiento de los conjuntos de datos sea más accesible.

[0008] En varias formas de realización, se puede usar un modelo de datos similar a la forma en la que el cerebro humano asocia y estructura la información. Usando este tipo de modelo de datos, la invención permite que los usuarios encuentren y entiendan la información pertinente en los conjuntos de datos de manera natural y permite así adquirir conocimiento a partir de los conjuntos de datos. Este tipo de modelo de datos se puede describir mediante las características siguientes:

1. cada concepto de información, tal y como se ha definido en la literatura y entendido por los expertos de dicho campo, se modela como un tipo de objetos,
2. cada objeto pertenece a un tipo de objetos,
3. cada objeto contiene suficiente información para que sea significativo y comprensible,
4. no se requiere que cada objeto dentro de un tipo de objetos contenga la misma clase de atributos o clasificación de información,
5. cada objeto tiene información acerca de con qué otros objetos está relacionado y qué representa cada relación.

[0009] Dado que el modelo de datos previamente descrito no impone ningún esquema rígido a la información, es capaz de manejar los cambios en la estructura de los conjuntos de datos con delicadeza. Asimismo, debido a que el modelo de datos se asemeja a la manera en la que el cerebro humano construye el conocimiento, resulta sencillo para los usuarios entender la información aunque no exista ningún esquema predefinido.

[0010] Un primer aspecto de la invención es un motor de base de datos, con estructuras de índice apropiadas, que permite que los conjuntos de datos que usan el modelo de datos descrito previamente se consulten y analicen de manera flexible, sencilla y receptiva. El método de búsqueda del motor de base de datos implica las siguientes etapas:

1. especificar, usando el modelo de datos, qué tipo de información debería recuperarse y qué criterio debería aplicarse,
2. enviar la especificación anterior al motor de base de datos,
3. recibir un resultado del motor de base de datos,
4. presentar el resultado recibido al usuario,

En varias formas de realización, las estructuras de índice están basadas en un mapa de bits.

[0011] Al usar el método descrito anteriormente para buscar el motor de base de datos, el usuario no necesita conocimientos acerca de cómo se estructuran internamente los conjuntos de datos en el motor de base de datos. En cambio, el usuario especifica qué tipo de información le interesa y qué criterio debe cumplir la información. Esto permite que el usuario se centre en la información y permite así adquirir conocimiento a partir de los conjuntos de datos.

[0012] Un segundo aspecto de la invención es la identificación de un tipo específico de preguntas, que con frecuencia se deben hacer a los conjuntos de datos, junto con un método y una terminología nuevos para especificar dicho tipo de pregunta. Este tipo específico de pregunta opera en relaciones indirectas en los conjuntos de datos y se puede usar para adquirir conocimientos que estarían ocultos si únicamente se usaran relaciones directas. Un ejemplo de una pregunta de este tipo podría ser: para un tipo específico de objeto, cómo se correlacionan los atributos de otros tipos de objetos. Otro ejemplo podría ser: para los objetos de un tipo de objetos específico, que cumplen un determinado criterio, qué atributos de otro tipo de objetos tienen en común. Estos tipos de preguntas son comunes en la consulta y el análisis de conjuntos de datos en los que un tipo de objetos específico se sitúa en el centro del análisis, por ejemplo un análisis céntrico de pacientes en la investigación clínica y la asistencia sanitaria. Este aspecto de la invención se puede usar junto con los otros aspectos de la invención, pero también se puede usar con otros motores de base de datos y modelos de datos.

[0013] Un tercer aspecto de la invención es una extensión del segundo aspecto de la invención que permite que dicho motor de base de datos maneje los tipos de preguntas anteriormente descritos de manera flexible, sencilla y receptiva. Este aspecto de la invención permite que se busquen estos tipos de preguntas utilizando el método de búsqueda tal y como se describe para el segundo aspecto de la invención.

[0014] Como resultado de los aspectos de la invención anteriormente descritos, la invención consigue el objetivo y permite específicamente acceder al conocimiento de los conjuntos de datos.

BREVE DESCRIPCIÓN DE LOS DIBUJOS

[0015] Con el objetivo de que se entienda fácilmente la manera según la cual se obtienen los objetos y ventajas previamente citados y otros de la invención, se proporcionará una descripción más particular de la invención brevemente descrita antes con referencia a formas de realización específicas de los mismos tal y como se ilustra en los dibujos anexos.

[0016] Teniendo en cuenta que estos dibujos únicamente representan formas de realización típicas de la invención y, por lo tanto, no deben considerarse como limitativos de su alcance, la invención se describirá y explicará con especificidad y detalle adicionales mediante los dibujos anexos, donde:

la fig. 1 es un diagrama que muestra un ejemplo de datos representados en el tipo de modelo de datos descrito,
la fig. 2 es un diagrama que muestra un ejemplo de una reflexión de un subconjunto de datos en objetos de tipo B,
la fig. 3 es un diagrama que muestra un ejemplo de cómo el índice de identidad y relación, utilizado por el motor de base de datos, puede buscar cada objeto en un conjunto de datos,
la fig. 4 es un diagrama que muestra un ejemplo de cómo el motor de base de datos puede usar el índice de relación para filtrar el conjunto de datos y encontrar toda información relacionada,
la fig. 5 es un diagrama que muestra un ejemplo de cómo el motor de base de datos puede reflejar un subconjunto de datos en objetos de tipo B,
la fig. 6 es un diagrama que muestra un ejemplo de cómo el motor de base de datos puede reflejar dos subconjuntos de datos usando dos puntos de reflexión diferentes,
la fig. 7 es un diagrama que muestra cómo se crea el índice de identidad, usado para el motor de base de datos,
la fig. 8 es un diagrama que muestra cómo se crea el índice de relación, usado para el motor de base de datos,
la fig. 9 es un diagrama que muestra cómo el motor de base de datos puede usar las estructuras de índice para responder a una consulta de búsqueda,
la fig. 10 es un diagrama que muestra cómo el motor de base de datos puede usar las estructuras de índice para responder a una consulta de búsqueda usando la lógica reflexiva,
la fig. 11 es un diagrama que muestra un ejemplo de datos con dos objetos raíz representados en el tipo de modelo de datos descrito,
la fig. 12 es un diagrama que muestra un ejemplo de cómo el índice de identidad, usado por el motor de base de datos, podría buscar cada objeto en un conjunto de datos particionado,
la fig. 13 es un diagrama que muestra cómo el motor de base de datos puede extender procesos previamente definidos cuando trabaja con datos particionados,
la fig. 14 es un diagrama que muestra un ejemplo de cómo el motor de base de datos puede actualizar las estructuras de índice cuando se añaden nuevos datos a un conjunto de datos,
la fig. 15 es un diagrama que muestra un ejemplo de cómo el motor de base de datos puede actualizar las estructuras de índice cuando se eliminan datos de un conjunto de datos,
la fig. 16 es un diagrama que muestra cómo se crea el índice de identidad para nuevos objetos cuando se añaden datos a un conjunto de datos,
la fig. 17 es un diagrama que muestra cómo se actualiza el índice de relación cuando se añaden nuevos datos a un conjunto de datos, y
la fig. 18 es un diagrama que muestra cómo se actualiza el índice de relación cuando se eliminan datos de un conjunto de datos.

DESCRIPCIÓN DETALLADA

[0017] La unidad más fundamental en el modelo de datos es un objeto. Cada objeto pertenece a un tipo de objetos que representa el concepto de la información contenida por el objeto. Los tipos de objetos se crean en función de cómo se define dicha información en la literatura y cómo la entienden los expertos de dicho campo. Con frecuencia, para reflejar cómo se comporta la información, no se requieren diferentes objetos del mismo tipo para contener el mismo tipo de información. Asimismo, cada objeto sabe con qué otros objetos está relacionado y qué significa dicha relación, es decir, la información representada por la presencia de la relación.

[0018] La fig. 1 proporciona un ejemplo gráfico de cómo se puede representar un conjunto de datos usando este modelo de datos. Los círculos representan objetos diferentes y las flechas representan relaciones dirigidas entre objetos diferentes. Una letra en un círculo indica el tipo del objeto y un número en el círculo es un identificador único de cada objeto. La fig. 1 muestra un objeto que pertenece al tipo de objeto A con un número identificador único 1, tres objetos que pertenecen al tipo de objeto B y se identifican por los números 2, 3 y 4, respectivamente, dos objetos que pertenecen al tipo de objeto C y se identifican por los números 5 y 6, respectivamente, y finalmente un objeto que pertenece al tipo de objeto D y se identifica por el número 7.

[0019] Tal y como se muestra en la figura, no se requiere que diferentes objetos del mismo tipo tengan las mismas relaciones. Del mismo modo, tal y como se muestra en la figura, los datos representados en este modelo se pueden representar fácilmente como un diagrama de flujo con relaciones dirigidas, que se asemeja a la forma en la que el cerebro humano asocia y estructura la información. En este tipo de modelo de datos, las actualizaciones frecuentes del conjunto de datos se manejan fácilmente, ya que cada objeto o cadena de objetos nuevo que se añade al conjunto de datos se añade como un subdiagrama de flujo en la posición apropiada en el

diagrama de flujo. Los cambios estructurales del conjunto de datos también se manejan fácilmente ya que cada tipo de objeto no impone una estructura rígida de la información contenida en el objeto.

[0020] La invención se destina específicamente a la identificación de un tipo específico de pregunta, que debe hacerse con frecuencia a conjuntos de datos, junto con un método y terminología nuevos para especificar este tipo de preguntas. Este tipo de preguntas se puede describir mediante las características siguientes:

1. tómese un subconjunto de datos
2. encontrar todos los objetos de un tipo específico que están incluidos en el subconjunto de datos o relacionados con él
3. crear un nuevo subconjunto de datos, a partir del conjunto de datos completo, que contenga los objetos que se han identificado en la etapa precedente, y todos los objetos relacionados con dichos objetos

[0021] La terminología utilizada para describir este tipo de preguntas se define mediante los conceptos siguientes:

• el método de consulta de un conjunto de datos de esta manera se denomina lógica reflexiva
• el tipo de objetos usado en la etapa 2 anterior se denomina el punto de reflexión
• el subconjunto de datos de la etapa 1 anterior se considera reflejado en el punto de reflexión durante la etapa 3

[0022] Se ha descubierto que la lógica reflexiva es una parte integral de cómo se necesita consultar y analizar los diferentes conjuntos de datos para acceder al conocimiento en una variedad de campos, que varían desde el académico a la investigación farmacéutica y la asistencia sanitaria. En un análisis de, por ejemplo, datos clínicos, un punto de reflexión típico sería el paciente, pero otros puntos de reflexión, por ejemplo, el estudio, también podrían ser de interés.

[0023] El ejemplo siguiente, de análisis de datos clínicos, puede mostrar cómo se puede usar la lógica reflexiva para adquirir conocimiento a partir de un conjunto de datos. Un investigador clínico puede estar interesado en hallar una explicación de por qué algunos pacientes a los que se les ha administrado una dosis de un fármaco tienen una alta presión sanguínea. Usando la terminología definida previamente, este problema se puede presentar como:

1. encontrar todas las mediciones de presión sanguínea que se consideran altas y que están relacionadas con los pacientes a los que se administró una dosis del fármaco,
2. reflejar los objetos que cumplen la condición precedente en el tipo de objetos que representan a los pacientes,
3. analizar el conjunto de datos resultante, que contiene toda la información de los pacientes pertinentes, con el objetivo de hallar una explicación para la presión sanguínea alta.

[0024] En el ejemplo anterior, el tipo de objetos que representa a los pacientes se usa como punto de reflexión y las mediciones de interés se reflejan en dicho punto de reflexión para obtener toda la información relacionada con dichos pacientes.

[0025] La fig. 2 muestra un ejemplo de cómo la lógica reflexiva puede transformar un conjunto de datos. En este ejemplo, un subconjunto de datos se refleja en objetos del tipo marcado como B. Una sección superior derecha de la figura muestra el subconjunto de datos seleccionado dentro de una línea continua. Una sección central izquierda de la figura muestra los objetos del tipo marcado como B indicados por una línea discontinua que están relacionados con el subconjunto de datos y que se usarán cuando se refleje el conjunto de datos. El lado inferior derecho de la figura muestra el resultado, indicado con una línea discontinua de raya y punto, del subconjunto reflejado en objetos del tipo marcado como B. Al interpretar los diferentes tipos de objetos de la siguiente manera, la fig. 2 se puede usar para describir el ejemplo anterior:

• los objetos del tipo marcado como A representan el estudio clínico llevado a cabo
• los objetos del tipo marcado como B representan a los pacientes
• los objetos del tipo marcado como C y D representan métodos diferentes para medir la presión sanguínea
• los objetos del tipo marcado como E representan mediciones de presión sanguínea,
• los objetos marcados con línea continua, es decir identificados por 10 y 12, representan mediciones de presión sanguínea alta,
• los objetos marcados con líneas discontinuas de rayas, es decir, los objetos identificados por 2 y 4, representan a los pacientes que tuvieron una medición de presión sanguínea alta, y
• los objetos marcados con líneas discontinuas de raya y punto representan objetos relacionados con los pacientes que tuvieron una medición de presión sanguínea alta.

[0026] La descripción y los ejemplos de lógica reflexiva se sirven del modelo de datos anteriormente descrito, pero la invención comprende además el uso de la lógica reflexiva en otros modelos de datos.

[0027] Tal y como se ha declarado anteriormente, un aspecto de la innovación es un método nuevo de indexar las relaciones y usar dichos índices para las preguntas ordinarias y la lógica reflexiva con el objetivo de permitir que un motor de base de datos utilice los aspectos de la invención definidos previamente.

[0028] Las estructuras de índice utilizadas se basan en un mapa de bits, lo que significa que se constituyen de vectores de bits donde cada bit puede ser cero o uno y cada bit se mapea a un significado específico. Además, la longitud del vector de bits, es decir, el número de bits del vector, siempre es un múltiplo de la longitud de palabra de la arquitectura del procesador dirigido. A modo de ejemplo, esto significa que, para una arquitectura de procesador de 64 bits, la longitud del vector de bits siempre será un múltiplo de 64. Este alineamiento de la longitud del vector de bits conlleva que las operaciones lógicas en los vectores de bits se puedan llevar a cabo usando operaciones a nivel de bits optimizadas, por ejemplo AND u OR. Sin embargo, para facilitar la comprensión del concepto de las estructuras de índice diferentes, los vectores de bits no han sido alineados a la longitud de palabra en los ejemplos usados más adelante.

[0029] Se crean un par de estructuras de índice de mapa de bits diferentes para que el motor de base de datos maneje tanto las preguntas ordinarias como la lógica reflexiva. Los dos estructuras de índice más destacadas de la invención son el índice de relación y el índice de identidad.

[0030] El índice de identidad consiste en vectores de índice que definen cada uno un objeto en un conjunto de datos. Cada vector de índice es un vector de bits en el que un bit se establece en uno y todos los demás bits se establecen en cero. De esta forma, el bit establecido en uno identificará únicamente el objeto del conjunto de datos. Dado que el índice de identidad usa un único bit para identificar cada objeto en un conjunto de datos, se pueden identificar múltiples objetos mediante un único vector de bits donde múltiples bits se establecen en uno. Dicho vector de bits se puede crear combinando múltiples vectores de índice de identidad utilizando operaciones OR a nivel de bits.

[0031] El índice de relación se constituye de vectores de índice que definen cada uno con qué objetos está relacionado un objeto específico. Cada vector de índice de relación contiene la combinación de todos los vectores de índice de identidad de los objetos con los que está relacionado el objeto actual. Esto significa que un vector de índice de relación es un vector de bits que identifica únicamente todos los objetos con los que está relacionado el objeto actual usando el esquema definido previamente.

[0032] Para cada tipo de objetos, se crea un vector de índice que consiste en la combinación de los vectores de índice de identidad de todos los objetos de este tipo. Basándose en estas estructuras de índice, es posible determinar qué objetos de un tipo específico están presentes en un subconjunto de datos.

[0033] La fig. 3 muestra un ejemplo de un índice de identidad y de relación para cada objeto en un conjunto de datos. La primera fila de ceros y unos es el índice de identidad y la segunda fila de ceros y unos es el índice de relación. Tal y como se puede observar en esta figura, cada objeto se identifica por exactamente un bit que se establece en uno en el vector de identidad. A modo de ejemplo, el vector de índice de identidad 0001000 identifica únicamente el objeto de tipo B identificado por el número 3. La figura muestra también que cada objeto del conjunto de datos tiene un vector de índice de identidad único. Esto permite que el índice de relación se interprete eficazmente como un vector de las identidades de todos los objetos con los que está relacionado el objeto actual.

[0034] En la forma de realización mostrada en la fig. 3, se le da al objeto 1 del tipo de objeto A el vector de índice de identidad 1000000 interpretado como un uno en una primera posición de bit y cero en todas las demás posiciones. El vector de índice tiene una longitud de 7 bits. Un primer objeto del tipo de objeto B se representa como 2 y se le da el vector de índice de identidad 0100000 y un segundo objeto del tipo de objeto B se representa como 3 y se le da el vector de índice de identidad 0001000. Los objetos restantes tienen vectores de índice de identidad correspondientes.

[0035] Dado que el índice de identidad usa un bit único para identificar cada objeto en un conjunto de datos, se pueden identificar múltiples objetos mediante un único vector de bits donde múltiples bits se establecen en uno. Este tipo de vector de bits se puede crear combinando múltiples vectores de índice de identidad utilizando operaciones OR a nivel de bits. Por ejemplo, en la fig. 3 el vector de bits 0101100 identificaría únicamente los objetos identificados por los números 2, 3 y 4.

[0036] El vector de índice de relación del objeto 1 (1000000) se determina mediante una combinación de los índices de identidad de todos los objetos que están relacionados con el objeto 1. El objeto 1 está relacionado con todos los demás objetos incluido él mismo. Como resultado de ello, cada posición del vector de índice de relación del objeto 1 se establecerá como uno porque cada objeto relacionado contribuirá con un uno de su vector de índice de identidad.

[0037] El objeto 4 (0000100) está relacionado con el objeto 1 (1000000), dando como resultado un uno en la primera posición del vector de índice de relación; con el objeto 6 (0000010), dando como resultado un uno en la

sexta posición; con el objeto 7 (0000001), dando como resultado un uno en la última posición; y consigo mismo (0000100) dando como resultado un uno en la quinta posición. Por tanto, el vector de índice de relación final del objeto 4 será 1000111 como se muestra en la fig. 3. Todos los demás vectores de índice de relación se forman de la misma manera. A título de ejemplo adicional, el objeto identificado por el número 2 tiene un vector de índice de relación de 1110000 que corresponde al vector de índice de identidad de los objetos identificados por los números 1, 2 y 5.

[0038] Al crear el índice de relación, resulta importante definir cuándo están relacionados dos objetos y cuándo no lo están. Por ejemplo, en la fig. 3, todos los objetos del tipo marcado como B están relacionados a través del objeto identificado por el número 1 y, sin embargo, los vectores de índice de relación para los objetos del tipo B no contienen esta relación. La razón de ello es que, en el modelo de datos usado en los ejemplos, se considera que un objeto está relacionado con sus padres e hijos y todos los objetos relacionados con estos excepto los hermanos del mismo tipo que el propio objeto. Esta definición de las relaciones se entiende más fácilmente si los diferentes objetos de la fig. 3 se interpretan como se expone a continuación:

- los objetos del tipo marcado como A representan un estudio clínico llevado a cabo
- los objetos del tipo marcado como B representan pacientes del estudio
- los objetos del tipo marcado como C representan la altura de un paciente
- los objetos del tipo marcado como D representan el peso de un paciente

[0039] Al usar la interpretación anterior, resulta evidente que los diferentes pacientes representados por los objetos identificados por los números 2, 3 y 4 no deberían estar directamente relacionados, ni la altura del paciente identificado por el número 2 debería estar relacionada con la altura del paciente identificado por el número 4. Sin embargo, la altura y el peso del paciente identificado por el número 4 deberían estar relacionados.

[0040] También se crea un tercer índice denominado índice identidad de tipo de objeto para que el motor de base de datos use los otros índices de manera más eficaz. Este índice consiste en un vector de índice para cada tipo de objetos, donde cada vector de índice se crea a partir de la combinación del vector de índice de identidad de todos los objetos de este tipo. Esto significa que un vector de índice de identidad de tipo de objeto es un vector de bits que identifica únicamente todos los objetos de un tipo específico usando el esquema definido previamente. Estas estructuras de índice se utilizan para permitir encontrar los objetos de un tipo específico que están presentes en un subconjunto de datos. A modo de ejemplo, en la fig. 3 el índice de identidad de tipo de objeto para los objetos de tipo C sería 0010010, que es una combinación de los vectores de índice de identidad de los dos objetos de tipo C.

[0041] Tal y como se ha indicado previamente, los vectores de bits usados para representar los diferentes vectores de índice se alinean con la longitud de palabra de la arquitectura del procesador dirigido. Esto se lleva a cabo mediante las etapas siguientes:

1. si el número de objetos es un múltiplo de la longitud de palabra, entonces no se requiere una acción adicional
2. si el número de objetos no es un múltiplo de la longitud de palabra, entonces se establece el número de bits como el múltiplo más bajo de la longitud de palabra que pueda representar todos los objetos y se establecen los bits adicionales en cero

[0042] Al usar el método de alineamiento anterior, la presencia de los bits adicionales añadidos no afectará a los índices, ya que un bit establecido en cero será tratado como si no hubiera un objeto presente.

[0043] El motor de base de datos utiliza las tres estructuras de índice descritas previamente para responder a las consultas de las siguientes características:

1. tómense uno o más criterios de filtración, es decir, restricciones en el conjunto de datos que se correlaciona con uno o más subconjuntos del conjunto de datos
2. tómese cómo debería combinarse el criterio de filtración
3. encontrar el subconjunto de objetos que cumplen cada criterio de filtración
4. encontrar todos los objetos que están incluidos en o relacionados con, directa o indirectamente, cada subconjunto
5. combinar los subconjuntos desde la etapa 4 utilizando operaciones de conjunto lógicas según la información de la etapa 2

[0044] A menudo se deben hacer preguntas del tipo descrito previamente a los conjuntos de datos para obtener conocimiento del conjunto de datos. A título de ejemplo, se considera un conjunto de datos de registros médicos donde se han registrado la altura y peso de pacientes. Un médico podría estar interesado en hallar toda la información relacionada con una medición de peso baja para averiguar si existe una necesidad de tomar acciones. La fig. 4 muestra un ejemplo de cómo el motor de base de datos puede utilizar el índice de identidad y

el índice de relación para responder a dicha consulta. Si la información de la figura se interpreta según lo que sigue, entonces el ejemplo anterior se puede usar para explicar adicionalmente la figura:

- los objetos del tipo marcado como A representan los hospitales que recogieron la información
- los objetos del tipo marcado como B representan a pacientes
- los objetos del tipo marcado como C representan la altura de un paciente
- los objetos del tipo marcado como D representan el peso de un paciente
- el objeto del tipo marcado como D que se identifica por el número 7 se considera una medición de peso baja
- se definen dos criterios de filtración por combinar usando AND
    - La información debería recogerse en un determinado hospital
    - La medición de peso debería considerarse baja

- los objetos marcados con una línea continua cumplen el primer criterio de filtración
- los objetos marcados con una línea discontinua de rayas cumplen el segundo criterio de filtración
- los objetos con línea discontinua de raya y punto son el resultado de la consulta

[0045] La fig. 4 muestra un ejemplo de cómo se pueden usar el índice de relación y el índice de identidad para encontrar todos los objetos que cumplen, directa o indirectamente, dos criterios. Más adelante se proporciona una descripción adicional en relación con la descripción de la figura 9. En la parte superior de la figura, los objetos dentro de una elipse de línea continua y una elipse de línea discontinua de rayas simbolizan los dos criterios. Un primer objeto (línea continua) tiene el vector de índice de relación 1111111 (objeto 1) y un segundo objeto (línea discontinua de rayas) tiene el vector de índice de relación 1000111 (objeto 7). En la parte mediana de la figura, los vectores de índice de relación de los dos conjuntos de objetos se combinan utilizando una operación AND a nivel de bits para producir un vector de índice de identidad resultante de 1000111 y, en la parte inferior de la figura, los objetos que coinciden con el vector de índice de identidad resultante se marcan con una línea discontinua de raya y punto. Estos objetos tienen vectores de índice de identidad 1000000 (objeto 1), 0000100 (objeto 4), 0000010 (objeto 6) y 0000001 (objeto 7), véase la tabla 1.

1000111	Vector de índice de identidad resultante
1000000	Objeto 1
0000100	Objeto 4
0000010	Objeto 6
0000001	Objeto 7

Tabla 1.

[0046] La fig. 5 muestra un ejemplo de cómo el motor de base de datos puede utilizar las estructuras de índice descritas previamente para responder a una pregunta que contiene lógica reflexiva. Si la información en la figura se interpreta según lo que sigue, el ejemplo anterior se puede usar para explicar adicionalmente la figura:

- los objetos del tipo marcado como A representan el estudio clínico llevado a cabo
- los objetos del tipo marcado como B representan a los pacientes
- los objetos del tipo marcado como C representan las mediciones de la presión sanguínea
- los objetos del tipo marcado como C que se identifican por los números 5 y 6 se consideran mediciones de presión sanguínea altas
- los objetos marcados con línea continua cumplen los criterios de filtración, en este caso mediciones de presión sanguínea consideradas altas
- los objetos marcados con línea discontinua de rayas representan objetos, del tipo de objetos en los que se refleja la consulta, que están relacionados con los objetos que cumplen los criterios de filtración
- los objetos marcados por una línea discontinua de raya y punto son el resultado de la consulta

[0047] En el lado superior de la figura, se identifican los objetos que cumplen los criterios de filtración. A continuación, el motor de base de datos combinará, utilizando operaciones OR a nivel de bits, los vectores de índice de relación de todos los objetos que cumplen los criterios de filtración. En este caso, el motor de base de datos combinará los dos vectores de índice de relación 1110000 y 1000110 usando una operación OR a nivel de bits para dar lugar a un nuevo vector de bits que será 1110110. A continuación, este vector de bits se combinará con el vector de índice de identidad de tipo de objeto para el punto de reflexión usando una operación AND a nivel de bits. El vector de bits resultante contendrá la identidad única de todos los objetos, del tipo de objetos en los que se refleja la consulta, que están relacionados con los objetos que cumplen los criterios de filtración.

[0048] En este caso, el vector de índice de identidad de tipo de objeto es 0101100 y se combinará, usando una operación AND a nivel de bits, con 1110110, lo que dará lugar al vector de bits resultante 0100100. Esta etapa se muestra en la parte central izquierda de la figura. En la etapa siguiente, el motor de base de datos usará el

vector de bits generado en la última etapa con el índice de identidad para encontrar los vectores de índice de relación de todos los objetos del tipo de objetos en los que se refleja la consulta. A continuación, el motor de base de datos combinará todo estos vectores de índice de relación usando operaciones OR a nivel de bits para recuperar un vector de bits que contenga la identidad única de todos los objetos que coinciden con la consulta.

[0049] En este caso, los vectores de índice de relación de los objetos, del tipo de objetos en los que se refleja la consulta, que están presentes en el último vector de bits son 1110000 y 1000111. La combinación de estos dos vectores de índice de relación usando una operación OR a nivel de bits produce un vector de bits de 1110111, que contiene la identidad única de todos los objetos que coinciden con la pregunta. Esta etapa se muestra en el lado de inferior izquierdo de la figura. La última etapa consiste en usar el índice de identidad para encontrar todos los objetos que están presentes en un último vector de bits. El resultado de la última etapa se muestra en el lado inferior de la figura.

[0050] En la sección central de la figura, los objetos del tipo marcado como B que coinciden con el vector de bits de resultado se marcan con líneas discontinuas de rayas. Dichos dos objetos, el objeto 2 (0100000) y el objeto 4 (0000100) se usarán mientras se refleje el conjunto de datos seleccionado. Un primer vector de índice de relación con el valor 1110000 (del objeto 2) se combina en una operación OR a nivel de bits con un segundo vector de índice de relación con el valor 1000111 (del objeto 4) para producir el último vector de bits 1110111. En el lado inferior derecho de la figura, los objetos que están presentes en el último vector de bits se marcan con una línea discontinua de raya y punto, véase la tabla 2.

1110111	Vector de índice de identidad resultante
1000000	Objeto 1
0100000	Objeto 2
0000100	Objeto 4
0010000	Objeto 5
0000010	Objeto 6
0000001	Objeto 7

Tabla 2.

[0051] En ocasiones se deben hacer preguntas que emplean la lógica reflexiva con múltiples puntos de reflexión a los conjuntos de datos para obtener un nivel de conocimiento más alto del conjunto de datos. A modo de ejemplo, se considera un conjunto de datos que contiene información de estudios clínicos donde algunos de los estudios han medido la presión sanguínea de los pacientes. Un investigador clínico podría estar interesado en observar pacientes con una alta presión sanguínea si formaran parte de un estudio en el que se ha administrado un determinado fármaco. Esta pregunta se puede formular, usando la terminología definida previamente para la lógica reflexiva, como:

1.
1. encontrar todos los objetos correlacionados con la administración del fármaco en cuestión
2. usar el tipo de objetos que representan estudios clínicos como el punto de reflexión y reflejar los objetos de la etapa 1 en este punto de reflexión
3. encontrar todos los objetos correlacionados con mediciones de presión sanguínea altas
4. usar el tipo de objetos que representa a pacientes como el punto de reflexión y reflejar los objetos de la etapa 3 en este punto de reflexión
5. encontrar todos los objetos que están presentes tanto en el resultado de la etapa 2 como el de la etapa 4
6. analizar la información de la etapa 5

Ny Fig. 6 (Fig. 7 i underlaget)

[0052] La fig. 6 muestra un ejemplo de cómo el motor de base de datos puede utilizar las estructuras de índice descritas previamente para responder a una pregunta como la anteriormente descrita. Si la información de la figura se interpreta según lo que sigue, el ejemplo anterior se puede usar para explicar adicionalmente la figura:

- los objetos del tipo marcado como A representan los estudios clínicos llevados a cabo
- los objetos del tipo marcado como B representan a los pacientes
- los objetos del tipo marcado como C representan las mediciones de presión sanguínea
- los objetos del tipo marcado como D representan la administración del fármaco en cuestión
- el objeto del tipo marcado como C, que se identifica por los números 6, se considera una medición de presión sanguínea alta
- los objetos marcados con línea continua forman parte de la consulta que se refleja en el tipo de objetos que representan los estudios clínicos

• los objetos marcados con línea discontinua de rayas forman parte de la consulta que se refleja en el tipo de objetos que representan a los pacientes
• los objetos marcados con línea discontinua de raya y punto son el resultado de la consulta
• el lado superior izquierdo de la figura identifica los objetos que coinciden con las dos partes de la consulta
• el lado superior derecho de la figura identifica los objetos que se usarán como puntos de reflexión
• el lado inferior derecho de la figura identifica los objetos que forman parte de los dos conjuntos, representando los dos partes de consulta, después de la reflexión
• el lado inferior izquierdo de la figura identifica el resultado de la consulta

[0053] En el lado superior izquierdo de la figura, se identifican los objetos que coinciden con las dos partes de la pregunta. Los objetos marcados con línea continua representan la administración del fármaco en cuestión y el objeto marcado con línea discontinua de rayas representa la medición de presión sanguínea alta. El motor de base de datos combinará en primer lugar, usando operaciones OR a nivel de bits, los vectores de índice de relación de todos los objetos que coinciden con cada parte de la consulta. En este caso, el motor de base de datos combinará los dos vectores de índice de relación 1110000 y 1000111 usando una operación OR a nivel de bits en un nuevo vector de bits que será 1110111. Este vector de bits se usa para la parte de la consulta que se reflejará en el tipo de objetos que representan los estudios clínicos.

[0054] Para la parte de la consulta que se reflejará en el tipo de objetos que representan a los pacientes, el motor de base de datos no tiene que combinar ningún vector de índice de relación, ya que únicamente un objeto cumple los criterios de filtración y dicho objeto tiene un vector de índice de relación de 1000111. En la siguiente etapa, el motor de base de datos combinará, usando operaciones AND a nivel de bits, los vectores de bits que se acaban de crear con los vectores de índice de identidad de tipo de objeto para el punto de reflexión correlacionado. En este caso, el vector de índice de identidad de tipo de objeto para el tipo de objetos que representan los estudios clínicos es 1000000 y el vector de índice de identidad de tipo de objeto para el tipo de objetos que representan a los pacientes es 0101100.

[0055] De este modo, el vector de bits 1110111 se combinará con 1000000 produciendo un vector de bits de 1000000 y el vector de bits 1000111 se combinará con 0101100 produciendo un vector de bits de 0000100. Los objetos correlacionados con los vectores de bits de la última etapa se muestran en el lado superior derecho de la figura. En la etapa siguiente, el motor de base de datos usará los vectores de bits generados por la última etapa con el índice de identidad para encontrar los vectores de índice de relación de todos los objetos del tipo de objetos en los que se refleja cada parte de consulta. A continuación, el motor de base de datos combinará todos estos vectores de índice de relación, para cada parte de consulta, usando operaciones OR nivel de bits para recuperar vectores de bits que contengan la identidad única de todos los objetos que coincidan con cada parte de pregunta.

[0056] En este caso un solo objeto, del tipo de objetos que representan los estudios clínicos, coincide con el vector de bits, de modo que no se necesita ninguna combinación. Este objeto tiene un vector de índice de relación de 1111111. Adicionalmente, un solo objeto, del tipo de objetos que representa a los pacientes, coincide con el vector de bits, de modo que no se necesita ninguna combinación. Este objeto tiene un vector de índice de relación de 1000111. Los objetos cuyos vectores de índice de identidad están presentes en estos dos vectores de bits se muestran en el lado inferior derecho de la figura.

[0057] En la última etapa, el motor de base de datos combinará los vectores de bits de la última etapa usando operaciones AND u OR a nivel de bits dependiendo de cómo se deban combinar las dos partes de consulta. En este caso, las partes de consulta deberían combinarse mediante una operación AND a nivel de bits ya que la pregunta se formuló como la combinación de las dos partes de consulta. Esto significa que los dos vectores de bits 1111111 y 1000111 se combinarán usando una operación AND a nivel de bits que produce el resultado 1000111. Este nuevo vector de bits contiene la identidad única de todos los objetos que coinciden con la consulta. El resultado de la consulta se muestra en el lado inferior izquierdo de la figura.

[0058] La fig. 7 es una representación esquemática de cómo el motor de base de datos construye los vectores de índice de identidad. En el bloque 10, se calcula el número total de objetos de la base de datos y en el bloque 12 se establece en 0 un contador de objetos. A continuación, se inicia un primer proceso de iteración en el bloque 14. Se lleva a cabo una pluralidad de etapas de proceso para o cada objeto raíz de la base de datos comenzando en el bloque 16 donde se crea un vector de índice de identidad en forma de un vector de bits. La longitud del vector de índice de identidad, es decir, el número de bits, se fija para que sea igual al número total de objetos de la base de datos tal como se ha calculado en el bloque 10.

[0059] En el bloque 18, se le da un valor al vector de índice de identidad. Todos los bits del vector se establecen en cero excepto un único bit que corresponde al valor actual del contador de objetos. Este único bit se establece en uno. De este modo, únicamente una posición de bit del vector de índice de identidad se establece en uno mientras que todos los demás bits se establecen en cero. A continuación, en el bloque 20, el contador de objetos se incrementa en uno. En el bloque 22 hay una pregunta que pregunta si el objeto actual del proceso de iteración tiene hijos. En tal caso, todas las etapas de iteración de los bloques 16 a 22 se repiten a continuación para cada

objeto hijo tal como se expone en el bloque 24. Al vector de índice de identidad de todos los objetos de la base de datos se le da un valor acorde a los objetos raíz. Si no existen hijos, el proceso de iteración continúa. Seguir el esquema para el conjunto de datos representado en la fig. 1 daría lugar a las etapas siguientes:

- calcular el número total de objetos en el conjunto de datos, en este caso 7 objetos

- crear un contador de objetos y establecer su valor en 0

- empezar por el primer objeto raíz, en este caso el objeto identificado por el número 1

- crear un vector de índice de identidad para el primer objeto raíz, en este caso el objeto identificado por el número 1

    - el vector de bits debería tener una longitud igual al número total de objetos, en este caso 7
    - establecer todos los bits en cero a excepción del bit con el mismo número que el contador de objetos, en este caso el número de bit 0
    - el vector de bits resultante es 1000000

- incrementar el contador de objetos en uno, en este caso el nuevo valor es 1

- dado que el primer objeto raíz contiene hijos, continuar el proceso para cada hijo

- crear un vector de índice de identidad para el primer hijo, en este caso el objeto identificado por el número 2

    - el vector de bits debería tener una longitud igual al número total de objetos, en este caso 7
    - establecer todos los bits en cero a excepción del bit con el mismo número que el contador de objetos, en este caso el número de bit 1
    - el vector de bits resultante es 0100000

- incrementar el contador de objetos en uno, en este caso el nuevo valor es 2

- dado que el objeto actual contiene hijos, continuar el proceso para cada hijo

- crear un vector de índice de identidad para el primer hijo, en este caso el objeto identificado por el número 5

    - el vector de bits debería tener una longitud igual al número total de objetos, en este caso 7
    - establecer todos los bits en cero a excepción del bit con el mismo número que el contador de objetos, en este caso el número de bit 2
    - el vector de bits resultante es 0010000

- incrementar el contador de objetos en uno, en este caso el nuevo valor es 3

- dado que el objeto actual no contiene hijos, continuar el proceso para el padre, en este caso el objeto identificado por el número 2

- dado que el objeto actual no contiene más hijos, continuar el proceso para el padre, en este caso el objeto identificado por el número 1

- crear un vector de índice de identidad para el segundo hijo, en este caso el objeto identificado por el número 3

    - el vector de bits debería tener una longitud igual al número total de objetos, en este caso 7
    - establecer todos los bits en cero a excepción del bit con el mismo número que el contador de objetos, en este caso el número de bit 3
    - el vector de bits resultante es 0001000

- incrementar el contador de objetos en uno, en este caso el nuevo valor es 4

- dado que el objeto actual no contiene hijos, continuar el proceso para el padre, en este caso el objeto identificado por el número 1

- crear un vector de índice de identidad para el tercer hijo, en este caso el objeto identificado por el número 4

    - el vector de bits debería tener una longitud igual al número total de objetos, en este caso 7
    - establecer todos los bits en cero a excepción del bit con el mismo número que el contador de objetos, en este caso el número de bit 4

◦ el vector de bits resultante es 0000100

• incrementar el contador de objetos en uno, en este caso el nuevo valor es 5

• dado que el objeto actual contiene hijos, continuar el proceso para cada hijo

• crear un vector de índice de identidad para el primer hijo, en este caso el objeto identificado por el número 6

◦ el vector de bits debería tener una longitud igual al número total de objetos, en este caso 7
◦ establecer todos los bits en cero a excepción del bit con el mismo número que el contador de objetos, en este caso el número de bit 5
◦ el vector de bits resultante es 0000010

• incrementar el contador de objetos en uno, en este caso el nuevo valor es 6

• dado que el objeto actual no contiene ningún hijo, continuar el proceso para el padre, en este caso el objeto identificado por el número 4

• crear un vector de índice de identidad para el segundo hijo, en este caso el objeto identificado por el número 7

◦ el vector de bits debería tener una longitud igual al número total de objetos, en este caso 7
◦ establecer todos los bits en cero a excepción del bit con el mismo número que el contador de objetos, en este caso el número de bit 6
◦ el vector de bits resultante es 0000001

• incrementar el contador de objetos en uno, en este caso el nuevo valor es 7

• dado que el objeto actual no contiene hijos, continuar el proceso para el padre, en este caso el objeto identificado por el número 4

• dado que el objeto actual no contiene más hijos, continuar el proceso para el padre, en este caso el objeto identificado por el número 1

• dado que el primer objeto raíz no contiene más hijos, continuar al siguiente objeto raíz

• dado que no existen más objetos raíz, el proceso está completo

[0060] La fig. 8 muestra cómo el motor de base de datos construye los vectores de índice de relación. El índice de relación se crea para todos los objetos de la base de datos empezado por el o los objeto(s) raíz. Un segundo proceso de iteración se inicia en el bloque 26. Se llevan a cabo una pluralidad de etapas del proceso para cada objeto raíz en la base de datos empezando en el bloque 28 donde se crea un vector de índice de relación. El vector de índice de relación es un vector de bits igual al vector de índice de identidad del objeto actual en el proceso de iteración.

[0061] El vector de índice de relación creado se completa en el bloque 30 con vectores de índice de identidad de los padres y todos los hijos del objeto actual en el proceso de iteración. Las operaciones OR nivel de bits se pueden usar cuando se agreguen los vectores de índice de identidad. En una etapa final como la que se muestra en el bloque 32, también los vectores de índice de identidad de todos los hermanos de diferentes tipos de objetos.

[0062] En el bloque 34 se hace una pregunta que pregunta si el objeto actual del proceso de iteración tiene hijos. En caso afirmativo, todas las etapas de iteración de los bloques 28 a 34 se repiten a continuación para cada objeto hijo, tal y como se expone en el bloque 36. Después de estas etapas, se les ha asignado a todos los objetos de la base de datos un valor de vector de índice de identidad único y un valor de vector de índice de relación. Así, los vectores de índice de relación se pueden usar para procesar la base de datos. Seguir este esquema para el conjunto de datos representado en la fig. 1 produciría las etapas siguientes y crearía los mismos vectores de índice de relación tal y como se muestra en la fig. 3:

• empezar por el primer objeto raíz, en este caso el objeto identificado por el número 1

• crear un vector de índice de relación para el primer objeto raíz, en este caso el objeto identificado por el número 1

◦ establecer el vector de índice de relación para que sea igual al vector de índice del objeto, en este caso 1000000

◦ usar operaciones OR a nivel de bits para añadir los vectores de índice de identidad para todos los padres y todos los hijos, en este caso 0100000, 0010000, 0001000, 0000100, 0000010 y 0000001
◦ usar operaciones OR a nivel de bits para añadir la identidad de todos los hermanos de diferentes tipos de objetos y todos los hermanos de diferentes clases de objetos para todos los padres, en este caso ninguno
◦ el vector de bits resultante es 1111111

• dado que el primer objeto raíz contiene hijos, continuar el proceso para cada hijo

• crear un vector de índice de relación para el primer objeto hijo, en este caso el objeto identificado por el número 2

◦ establecer el vector de índice de relación para que sea igual al vector de índice del objeto, en este caso 0100000
◦ usar operaciones OR a nivel de bits para añadir los vectores de índice de identidad para todos los padres y todos los hijos, en este caso 1000000 y 0010000
◦ usar operaciones OR a nivel de bits para añadir la identidad de todos los hermanos de diferentes tipos de objetos y todos los hermanos de diferentes clases de objetos para todos los padres, en este caso ninguno
◦ el vector de bits resultante es 1110000

• dado que el objeto actual contiene hijos, continuar el proceso para cada hijo

• crear un vector de índice de relación para el primer objeto hijo, en este caso el objeto identificado por el número 5

◦ establecer el vector de índice de relación para que sea igual al vector de índice del objeto, en este caso 0010000
◦ usar operaciones OR a nivel de bits para añadir los vectores de índice de identidad para todos los padres y todos los hijos, en este caso 1000000 y 0100000
◦ usar operaciones OR a nivel de bits para añadir la identidad de todos los hermanos de diferentes tipos de objetos y todos los hermanos de diferentes clases de objetos para todos los padres, en este caso ninguno
◦ el vector de bits resultante es 1110000

• dado que el objeto actual no contiene ningún hijo, continuar el proceso para el padre, en este caso el objeto identificado por el número 2

• dado que el objeto actual no contiene más hijos, continuar el proceso para el padre, en este caso el objeto identificado por el número 1

• crear un vector de índice de relación para el segundo objeto hijo, en este caso el objeto identificado por el número 3

◦ establecer el vector de índice de relación para que sea igual al vector de índice del objeto, en este caso 0001000
◦ usar operaciones OR a nivel de bits para añadir los vectores de índice de identidad para todos los padres y todos los hijos, en este caso 1000000
◦ usar operaciones OR a nivel de bits para añadir la identidad de todos los hermanos de diferentes tipos de objetos y todos los hermanos de diferentes clases de objetos para todos los padres, en este caso ninguno
◦ el vector de bits resultante es 1001000

• dado que el objeto actual no contiene ningún hijo, continuar el proceso para el padre, en este caso el objeto identificado por el número 1

• crear un vector de índice de relación para el tercer objeto hijo, en este caso el objeto identificado por el número 4

◦ establecer el vector de índice de relación para que sea igual al vector de índice del objeto, en este caso 0000100
◦ usar operaciones OR a nivel de bits para añadir los vectores de índice de identidad para todos los padres y todos los hijos, en este caso 1000000, 0000010 y 0000001
◦ usar operaciones OR a nivel de bits para añadir la identidad de todos los hermanos de diferentes tipos de objetos y todos los hermanos de diferentes clases de objetos para todos los padres, en este caso ninguno
◦ el vector de bits resultante es 1000111

• dado que el objeto actual contiene hijos, continuar el proceso para cada hijo

• crear un vector de índice de relación para el primer objeto hijo, en este caso el objeto identificado por el número 6

◦ establecer el vector de índice de relación para que sea igual al vector de índice del objeto, en este caso 0000010
◦ usar operaciones OR a nivel de bits para añadir los vectores de índice de identidad para todos los padres y todos los hijos, en este caso 1000000 y 0000100
◦ usar operaciones OR a nivel de bits para añadir la identidad de todos los hermanos de diferentes tipos de objetos y todos los hermanos de diferentes clases de objetos para todos los padres, en este caso 0000001
◦ el vector de bits resultante es 1000111

• dado que el objeto actual no contiene ningún hijo, continuar el proceso para el padre, en este caso el objeto identificado por el número 4

• crear un vector de índice de relación para el primer objeto hijo, en este caso el objeto identificado por el número 7

◦ establecer el vector de índice de relación para que sea igual al vector de índice del objeto, en este caso 0000001
◦ usar operaciones OR a nivel de bits para añadir los vectores de índice de identidad para todos los padres y todos los hijos, en este caso 1000000 y 0000100
◦ usar operaciones OR a nivel de bits para añadir la identidad de todos los hermanos de diferentes tipos de objetos y todos los hermanos de diferentes clases de objetos para todos los padres, en este caso 0000010
◦ el vector de bits resultante es 1000111

• dado que el objeto actual no contiene hijos, continuar el proceso para el padre, en este caso el objeto identificado por el número 4

• dado que el objeto actual no contiene más hijos, continuar el proceso para el padre, en este caso el objeto identificado por el número 1

• dado que el primer objeto raíz no contiene más hijos, continuar al siguiente objeto raíz

• dado que no existen más objetos raíz, el proceso está completo

[0063] La fig. 9 muestra cómo el motor de base de datos utiliza las diferentes estructuras de índice para responder a una consulta como la anteriormente descrita. Seguir este esquema para el ejemplo anterior y el conjunto de datos representado en la fig. 4 produciría las etapas siguientes:

• crear un vector de resultado como un vector de bits de longitud igual a los vectores de índice de identidad

• para los primeros criterios de filtración, encontrar todos los objetos que cumplan el criterio, en este caso el objeto identificado por el número 1

◦ combinar los vectores de índice de relación de todos los objetos encontrados previamente usando operaciones OR a nivel de bits, en este caso el vector de índice de relación es 1111111 y no se requiere ninguna combinación
◦ añadir el vector de índice de relación combinado de la etapa precedente al vector de resultado usando operaciones AND u OR a nivel de bits dependiendo de cómo se deberían combinar los criterios

▪ el vector de resultado actual es 0000000
▪ el vector de índice de relación combinado es 1111111
▪ el criterio debería añadirse usando OR, ya que es el primer criterio
▪ el resultado es 1111111

• para el segundo criterio de filtración, encontrar todos los objetos que cumplen el criterio, en este caso el objeto identificado por el número 7

◦ combinar los vectores de índice de relación de todos los objetos encontrados previamente usando operaciones OR a nivel de bits, en este caso el vector de índice de relación es 1000111 y no se necesita ninguna combinación
◦ añadir el vector de índice de relación combinado de la etapa precedente al vector de resultado usando AND u OR a nivel de bits dependiendo de cómo se deberían combinar los criterios

▪ el vector de resultado actual es 1111111
▪ el vector de índice de relación combinado es 1000111

▪ el criterio se debería añadir usando AND, ya que el primer y el segundo criterio se han definido para que se combinen usando AND
▪ el resultado es 1000111

• dado que no existen más criterios de filtración, devolver todos los objetos cuyo vector de índice de identidad esté presente en el vector de resultado, en este caso los objetos identificados por los números 1, 4, 6 y 7 tal y como se muestra en la parte inferior de la figura 4

[0064] En el bloque 38 se crea un vector de resultado. El vector de resultado comienza con un valor de cero. La longitud del vector de resultado es igual a la longitud o los vectores de índice de identidad. A continuación, se inicia un proceso de iteración para cada criterio de filtración en el bloque 40. Una primera fase en este proceso de iteración es encontrar todos los objetos que cumplen el criterio, tal y como se muestra en el bloque 42. En el bloque 44, todos los vectores de índice de relación de los objetos encontrados se combinan en un vector de índice de relación combinado usando operaciones OR lógicas. El vector de índice de relación combinado se añade en el bloque 46 al vector de resultado creado usando operaciones AND u OR lógicas apropiadas dependiendo de los criterios de consulta o filtración. El proceso de iteración se completa cuando todos los criterios se han añadido al vector de resultado, tal y como se muestra en el bloque 48. Finalmente, en el bloque 50, todos los objetos con vectores de índice de identidad presentes en el vector de resultado se devuelven.

[0065] La parte más diferenciadora de la innovación es cómo el motor de base de datos, puede utilizar de forma eficiente las estructuras de índice anteriormente descritas para responder a preguntas que contienen lógica reflexiva con un punto de reflexión arbitrario. Este tipo de preguntas se puede describir mediante las siguientes características usando la terminología de la lógica reflexiva definida previamente:

1. tómense uno o más criterios de filtración, es decir, restricciones en el conjunto de datos que se correlaciona con uno o más subconjuntos del conjunto de datos
2. tómese cómo debería combinarse el criterio de filtración
3. tómese un tipo de objetos en el que la consulta debería reflejarse
4. encontrar el subconjunto de objetos que cumplen cada criterio de filtración
5. encontrar todos los objetos que están incluidos en o relacionados con, directa o indirectamente, cada subconjunto de la etapa 4
6. combinar los subconjuntos de la etapa 5 usando operaciones de conjunto lógicas según la información de la etapa 2
7. encontrar todos los objetos del tipo de objetos especificados en la etapa 3 que están presentes en el subconjunto de la etapa 6
8. para cada objeto de la etapa 7, encontrar todos los objetos con los que está relacionado, directa o indirectamente
9. combinar todos los objetos de la etapa 8, incluidos los objetos de la etapa 7, en un subconjunto nuevo

[0066] Con frecuencia, es necesario hacer las preguntas del tipo previamente descrito a los conjuntos de datos para obtener un nivel superior de conocimiento del conjunto de datos. A título de ejemplo, se considera un conjunto de datos que contiene información de un estudio clínico en el que se ha medido la presión sanguínea de los pacientes. Un investigador clínico podría interesarse por encontrar una explicación de por qué determinados pacientes tienen una presión sanguínea alta. Esta pregunta se puede formular usando la terminología definida previamente para la lógica reflexiva como sigue:

1. encontrar todos los objetos correlacionados con mediciones de presión sanguínea altas
2. usar el tipo de objetos que representan a los pacientes como el punto de reflexión y reflejar los objetos de la etapa 1 en este punto de reflexión
3. analizar la información de la etapa 2 con el objetivo de encontrar una explicación para la presión sanguínea alta

Fig. 10 (Fig. 11 i underlag)

[0067] La fig. 10 muestra cómo el motor de base de datos utiliza las diferentes estructuras de índice para responder a una consulta de lógica reflexiva tal como la que se ha descrito previamente. En primer lugar, se crea un vector de resultado vacío en el bloque 52 y se inicia un proceso de iteración adicional en el bloque 54 para cada criterio de filtración. El vector de resultado se inicia con un valor de cero. La longitud del vector de resultado es igual a la longitud o los vectores de índice de identidad. Todos los objetos que cumplen el criterio se determinan en el bloque 56. A continuación, en el bloque 58 se combinan todos los vectores de índice de relación usando operaciones OR lógicas para producir un vector de índice de relación combinado. En el bloque 60, se añade el vector de índice de relación combinado al vector de resultado dependiendo de la combinación de los criterios de consulta o de filtración. El proceso de iteración se completa cuando todos los criterios se han añadido al vector de resultado tal y como se muestra en el bloque 62. Posteriormente, encontrar todos los objetos, del tipo de objetos en los que debería reflejarse la consulta, que están presentes en el vector de resultado, bloque 64.

[0068] A continuación, la reflexión tiene lugar en el bloque 66 con la etapa de creación de un vector de resultado de reflexión de misma longitud que los vectores de índice de identidad y, en un proceso de iteración para cada una objeto, bloque 68, continuar con la etapa la etapa de adición del vector de índice de relación al vector de resultado de reflexión usando una operación OR a nivel de bits en el bloque 70. Cuando todos los vectores de índice han sido añadidos al vector de resultado de reflexión tal y como se muestra en el bloque 72, el proceso de iteración se completa y finalmente, en el bloque el proceso se completa en el bloque 74 con la etapa de devolución de todos los objetos con vectores de índice de identidad presentes en el vector de resultado de reflexión.

[0069] Seguir este esquema para el ejemplo anterior y el conjunto de datos representado en la fig. 5 produciría las etapas siguientes:

- crear un vector de resultado como un vector de bits de misma longitud que los vectores de índice de identidad
- para el primer criterio de filtración, encontrar todos los objetos que cumplen el criterio, en este caso los objetos identificados por los números 5 y 6
    - combinar los vectores de índice de relación de todos los objetos encontrados previamente usando operaciones OR a nivel de bits, en este caso los vectores de índice de relación son 1110000 y 1000110 y el resultado combinado es 1110110
    - añadir el vector de índice de relación combinado de la etapa anterior al vector de resultado usando operaciones AND u OR a nivel de bits dependiendo de cómo debería combinarse el criterio
        - el vector de resultado actual es 0000000
        - el vector de índice de relación combinado es 1110110
        - el criterio debería añadirse usando OR, ya que es el primer criterio
        - el resultado es 1110110
- dado que no existen más criterios de filtración, continuar a la etapa siguiente
- encontrar todos los objetos, del tipo de objetos en los que se debería reflejar la consulta, que están presentes en el vector de resultado
    - el vector de resultado es 1110110
    - el vector de índice de identidad de tipo de objeto, del tipo de objetos en los que se refleja la consulta, es 0101100
    - la combinación, usando una operación AND a nivel de bits, de los dos vectores de bits anteriores es 0100100
    - el vector de bits anterior contiene la identidad única de los objetos identificados por los números 2 y 4
- crear un vector de resultado de reflexión como un vector de bits de misma longitud que los vectores de índice de identidad
- para el primer objeto que identificado previamente, añadir el vector de índice de relación al vector de resultado de reflexión usando una operación OR a nivel de bits
    - el primer objeto identificado es el objeto identificado por el número 2
    - el vector de índice de relación de dicho objeto es 1110000
    - el vector de resultado de reflexión actual es 0000000
    - el resultado es 1110000
- para el segundo objeto identificado previamente, añadir el vector de índice de relación al vector de resultado de reflexión usando una operación OR a nivel de bits
    - el segundo objeto identificado es el objeto identificado por el número 4
    - el vector de índice de relación de dicho objeto es 1000111
    - el vector de resultado de reflexión actual es 1110000
    - el resultado es 1110111
- dado que no se han identificado más objetos previamente, devolver todos los objetos cuyo vector de índice de identidad está presente en el vector de resultado de reflexión, en este caso los objetos identificados por los números 1, 2, 4, 5, 6 y 7

[0070] Todas las estructuras de índice anteriormente descritas se encuentran en la forma más básica, pero la invención incluye también optimizaciones tales como el uso de compresión y codificación más complejas. Se puede aumentar la eficiencia espacial y computacional de las estructuras de índice usando un algoritmo de compresión basado en la codificación por longitud de secuencia (RLE, por sus siglas en inglés). Un algoritmo de compresión basado en la RLE comprime secuencias de ceros o unos consecutivas en un vector de índice. Esto

funciona excepcionalmente bien en los vectores de índice descritos por la invención debido a la forma en la que se han creado. Todos los vectores de índice de identidad tienen un bit establecido en uno y todos los demás bits establecidos en cero. Dado que un algoritmo de compresión basado en la RLE comprime secuencias consecutivas de ceros, los vectores de índice de identidad se pueden comprimir significativamente.

[0071] Los bits de los vectores de índice de relación que corresponden a las identidades de todos los objetos con las que el objeto actual está relacionado están establecidos en uno y todos los demás bits están establecidos en cero. Dado que los vectores de índice de identidad se crean recorriendo el modelo de datos como un recorrido de árbol en profundidad, es altamente probable que los bits que corresponden a todos los objetos relacionados sean una o más secuencias de unos consecutivas. Dado que un algoritmo de compresión basado en la RLE comprime secuencias de ceros y unos consecutivas, los vectores de índice de relación pueden comprimirse significativamente.

[0072] La invención permite que los datos se particionen en diferentes particiones que se pueden indexar y manejar por separado usando los mismos procesos que en el indexado de los datos usando el método anteriormente descrito. Esto permite que cada partición se actualice, indexe y procese independientemente. Dado que cada partición se puede procesar independientemente, esto también permite que cada partición de se procese en paralelo.

[0073] La partición más básica se lleva a cabo usando una partición para cada objeto del tipo de objetos usados para el objeto raíz del modelo de datos. La fig. 11 proporciona un ejemplo gráfico de cómo se puede representar un conjunto de datos con múltiples objetos raíz. En esta figura, dos objetos raíz diferentes, los objetos identificados por los números 1 y 8, se muestran como estructuras de árbol diferentes. Si estos datos se particionan usando este esquema, se crearán dos particiones diferentes. La primera partición contendrá el árbol en el lado izquierdo de la figura, con un objeto raíz identificado por el número 1, y la segunda partición contendrá el árbol en el lado derecho de la figura, con un objeto raíz identificado por el número 8.

[0074] Para permitir que cada partición se indexe y procese independientemente, los vectores de índice de particiones diferentes están completamente separados.

[0075] La fig. 12 muestra un ejemplo de un índice de identidad para un conjunto de datos pequeño con dos particiones diferentes, que están marcadas con una caja en línea continua. Cada partición se identifica por un número en el lado superior izquierdo de la caja y el vector de índice de identidad se imprime junto a cada objeto. Por ejemplo, el objeto identificado por el número 5 pertenece a la partición identificada por el número 1 y tiene un vector de índice de identidad o 0001000. El objeto identificado por el número 11 pertenece a la partición identificada por el número 2 y tiene un vector de índice de identidad de 001000.

[0076] La fig. 13 muestra cómo el motor de base de datos utiliza los procesos previamente descritos cuando los datos se dividen en particiones diferentes. Esta forma genérica de extender un proceso se utiliza para crear índices y para responder a consultas. Seguir este esquema usando el proceso de creación de índice de identidad de la fig. 7 como el subproceso y usando el conjunto de datos representado en la fig. 12, véase más arriba, produciría las etapas siguientes:

• dado que el proceso de creación de índice de identidad no devuelve un resultado, no debería crearse ningún contenedor de resultado

• para la primera partición, en este caso la partición identificada por el número 1, ejecutar el proceso de creación de índice de identidad

  ◦ calcular el número total de objetos en la partición, en este caso 7 objetos
  ◦ crear un contador de objetos y establecer su valor en 0
  ◦ empezar por el primer objeto raíz, en este caso el objeto identificado por el número 1
  ◦ crear un vector de índice de identidad para el primer objeto raíz, en este caso el objeto identificado por el número 1

    ▪ el vector de bits debería ser de longitud igual al número total de objetos, en este caso 7
    ▪ establecer todos los bits en cero excepto el bit de mismo número que el contador de objetos, en este caso el número de bit 0
    ▪ el vector de bits resultante es 1000000

  ◦ incrementar en uno el contador de objetos, en este caso el valor nuevo es 1
  ◦ dado que el primer objeto raíz contiene hijos, continuar el proceso para cada hijo
  ◦ crear un vector de índice de identidad para el primer hijo, en este caso el objeto identificado por el número 2

    ▪ el vector de bits debería ser de longitud igual al número total de objetos, en este caso 7

▪ establecer todos los bits en cero a excepción del bit de mismo número que el contador de objetos, en este caso el número de bit 1
▪ el vector de bits resultante es 0100000

◦ incrementar en uno el contador de objetos, en este caso el valor nuevo es 2
◦ dado que el objeto actual contiene hijos, continuar el proceso para cada hijo
◦ crear un vector de índice de identidad para el primer hijo, en este caso el objeto identificado por el número 4

▪ el vector de bits debería tener una longitud igual al número total de objetos, en este caso 7
▪ establecer todos los bits en cero a excepción del bit de mismo número que el contador de objetos, en este caso el número de bit 2
▪ el vector de bits resultante es 0010000

◦ incrementar en uno el contador de objetos, en este caso el valor nuevo es 3
◦ dado que el objeto actual no contiene ningún hijo, continuar el proceso para el padre, en este caso el objeto identificado por el número 2
◦ crear un vector de índice de identidad para el segundo hijo, en este caso el objeto identificado por el número 5

▪ el vector de bits debería tener una longitud igual al número total de objetos, en este caso 7
▪ establecer todos los bits en cero a excepción del bit de mismo número que el contador de objetos, en este caso el número de bit 3
▪ el vector de bits resultante es 0001000

◦ incrementar en uno el contador de objetos, en este caso el valor nuevo es 4
◦ dado que el objeto actual no contiene hijos, continuar el proceso para el padre, en este caso el objeto identificado por el número 2
◦ dado que el objeto actual no contiene más hijos, continuar el proceso para el padre, en este caso el objeto identificado por el número 1
◦ crear un vector de índice de identidad para el segundo hijo, en este caso el objeto identificado por el número 3

▪ el vector de bits debería tener una longitud igual al número total de objetos, en este caso 7
▪ establecer todos los bits en cero a excepción del bit de mismo número que el contador de objetos, en este caso el número de bit 4
▪ el vector de bits resultante es 0000100

◦ incrementar en uno el contador de objetos, en este caso el valor nuevo es 5
◦ dado que el objeto actual contiene hijos, continuar el proceso para cada hijo
◦ crear un vector de índice de identidad para el primer hijo, en este caso el objeto identificado por el número 6

▪ el vector de bits debería tener una longitud igual al número total de objetos, en este caso 7
▪ establecer todos los bits en cero a excepción del bit de mismo número que el contador de objetos, en este caso el número de bit 5
▪ el vector de bits resultante es 0000010

◦ incrementar en uno el contador de objetos, en este caso el valor nuevo es 6
◦ dado que el objeto actual no contiene ningún hijo, continuar el proceso para el padre, en este caso el objeto identificado por el número 3
◦ crear un vector de índice de identidad para el segundo hijo, en este caso el objeto identificado por el número 7

▪ el vector de bits debería tener una longitud igual al número total de objetos, en este caso 7
▪ establecer todos los bits en cero a excepción del bit de mismo número que el contador de objetos, en este caso el número de bit 6
▪ el vector de bits resultante es 0000001

◦ incrementar en uno el contador de objetos, en este caso el valor nuevo es 7
◦ dado que el objeto actual no contiene ningún hijo, continuar el proceso para el padre, en este caso el objeto identificado por el número 3
◦ dado que el objeto actual no contiene más hijos, continuar el proceso para el padre, en este caso el objeto identificado por el número 1
◦ dado que el primer objeto raíz no contiene más hijos, continuar al objeto raíz siguiente
◦ dado que no existen más objetos raíz, el subproceso está completo

• dado que el subproceso no devuelve un resultado, no se añade nada al contenedor de resultado

• para la segunda partición, en este caso la partición identificada por el número 2, ejecutar el proceso de creación de índice de identidad

◦ calcular el número total de objetos en la partición, en este caso 6 objetos
◦ crear un contador de objetos y establecer su valor en 0
◦ empezar por el primer objeto raíz, en este caso el objeto identificado por el número 8
◦ crear un vector de índice de identidad para el primer objeto raíz, en este caso el objeto identificado por el número 8

▪ el vector de bits debería tener una longitud igual al número total de objetos, en este caso 6
▪ establecer todos los bits en cero a excepción del bit de mismo número que el contador de objetos, en este caso el número de bit 0
▪ el vector de bits resultante es 100000

◦ incrementar en uno el contador de objetos, en este caso el valor nuevo es 1
◦ dado que el primer objeto raíz contiene hijos, continuar el proceso para cada hijo
◦ crear un vector de índice de identidad para el primer hijo, en este caso el objeto identificado por el número 9

▪ el vector de bits debería tener una longitud igual al número total de objetos, en este caso 6
▪ establecer todos los bits en cero a excepción del bit de mismo número que el contador de objetos, en este caso el número de bit 1
▪ el vector de bits resultante es 010000

◦ incrementar en uno el contador de objetos, en este caso el valor nuevo es 2
◦ dado que el objeto actual no contiene hijos, continuar el proceso para cada hijo
◦ crear un vector de índice de identidad para el primer hijo, en este caso el objeto identificado por el número 11

▪ el vector de bits debería tener una longitud igual al número total de objetos, en este caso 6
▪ establecer todos los bits en cero a excepción del bit de mismo número que el contador de objetos, en este caso el número de bit 2
▪ el vector de bits resultante es 001000

◦ incrementar en uno el contador de objetos, en este caso el valor nuevo es 3
◦ dado que el objeto actual no contiene hijos, continuar el proceso para el padre, en este caso el objeto identificado por el número 9
◦ dado que el objeto actual no contiene más hijos, continuar el proceso para el padre, en este caso el objeto identificado por el número 8
◦ crear un vector de índice de identidad para el segundo hijo, en este caso el objeto identificado por el número 10

▪ el vector de bits debería tener una longitud igual al número total de objetos, en este caso 6
▪ establecer todos los bits en cero a excepción del bit de mismo número que el contador de objetos, en este caso el número de bit 3
▪ el vector de bits resultante es 000100

◦ incrementar en uno el contador de objetos, en este caso el valor nuevo es 4
◦ dado que el objeto actual contiene hijos, continuar el proceso para cada hijo
◦ crear un vector de índice de identidad para el primer hijo, en este caso el objeto identificado por el número 12

▪ el vector de bits debería tener una longitud igual al número total de objetos, en este caso 6
▪ establecer todos los bits en cero a excepción del bit de mismo número que el contador de objetos, en este caso el número de bit 4
▪ el vector de bits resultante es 000010

◦ incrementar en uno el contador de objetos, en este caso el valor nuevo es 5
◦ dado que el objeto actual no contiene hijos, continuar el proceso para el padre, en este caso el objeto identificado por el número 10
◦ crear un vector de índice de identidad para el segundo hijo, en este caso el objeto identificado por el número 13

▪ el vector de bits debería tener una longitud igual al número total de objetos, en este caso 6

▪ establecer todos los bits en cero a excepción del bit de mismo número que el contador de objetos, en este caso el número de bit 5
▪ el vector de bits resultante es 000001

◦ incrementar en uno el contador de objetos, en este caso el valor nuevo es 6
◦ dado que el objeto actual no contiene hijos, continuar el proceso para el padre, en este caso el objeto identificado por el número 10
◦ dado que el objeto actual no contiene más hijos, continuar el proceso para el padre, en este caso el objeto identificado por el número 8
◦ dado que el primer objeto raíz no contiene más hijos, continuar al objeto raíz siguiente
◦ dado que no existen más objetos raíz, el subproceso está completo

• dado que el subproceso no devuelve un resultado, no se añade nada al contenedor de resultado

• dado que no existen más particiones, el proceso está completo

[0077] Los procesos mostrados en la fig. 13 comprenden las etapas siguientes: crear un contenedor de resultado, bloque 66; para cada partición, bloque 68; ejecutar un subproceso, bloque 70; si el subproceso devuelve un resultado, añadir el resultado al contenedor de resultado, bloque 72; finalizarlo todo cuando todos los subprocesos hayan sido ejecutados, bloque 74; y devolver finalmente un contenedor de resultado, bloque 76.

[0078] Los tipos de actualizaciones que se pueden llevar a cabo en un conjunto de datos se pueden dividir en cuatro categorías diferentes: adición de objetos nuevos, eliminación de objetos existentes, actualizaciones de la información en los objetos existentes y cambios en las relaciones entre objetos. Dado que las actualizaciones en la información en los objetos existentes no modifican las relaciones o la identidad del objeto, no son necesarias actualizaciones en las estructuras de índice. Para las otras categorías de actualizaciones de conjuntos de datos, las estructuras de índice deben modificarse para ser válidas para el conjunto de datos modificado.

[0079] La fig. 14 muestra un ejemplo de cómo el motor de base de datos puede actualizar las estructuras de índice cuando se añade un objeto nuevo al conjunto de datos. La información de la figura debería interpretarse como sigue:

• el lado superior izquierdo de la figura muestra el conjunto de datos original
• el lado superior derecho de la figura muestra el conjunto de datos con el objeto nuevo, que está marcado con una línea continua
• el lado inferior derecho de la figura muestra el conjunto de datos con estructuras de índice actualizadas
• todos los vectores de índice que han sido modificados están cursiva, todos los demás vectores de índice no han sido modificados

[0080] En este ejemplo un objeto, identificado por el número 8, se añade al conjunto de datos como un hijo del objeto identificado por el número 3. Para mantener el índice de estructuras válido, el motor de base de datos debe actualizar el índice de identidad, el índice de relación y el índice de identidad de tipo de objeto. El motor de base de datos se inicia creando un vector de índice de identidad nuevo para el objeto identificado por el número 8. En este caso, el vector de índice de identidad nuevo será 00000001. La etapa siguiente, para el motor de base de datos, es crear un vector de índice de relación para el objeto nuevo. En este caso, el vector de índice de relación nuevo es 10010001. En la etapa siguiente, los vectores de índice de relación de todos los objetos, que están relacionados con el objeto nuevo, se actualizan. En este caso, el vector de índice de relación del objeto identificado por el número 1 se actualiza a 11111111 y el vector de índice de relación del objeto identificado por el número 3 se actualiza a 10010001. La última etapa consiste en que el motor de base de datos actualice el índice de identidad de tipo de objeto. En este caso el vector de índice de identidad del nuevo objeto se añade al vector de índice de identidad de tipo de objeto para el tipo de objetos marcado como D, lo que tiene como resultado un vector nuevo de 00000011.

[0081] Como se puede observar en la fig. 14, la longitud de los vectores de índice nuevos y actualizados es un bit más larga que la de los otros vectores de índice. Debido a la forma en la que funcionan las operaciones a nivel de bits, las operaciones en los vectores de bits seguirán trabajando como se había previsto. La razón de ello es que el vector de bits más corto se rellenará implícitamente con ceros para alcanzar la misma longitud que el vector de bits más largo antes de que se lleve a cabo la operación a nivel de bits.

[0082] La fig. 15 muestra un ejemplo de cómo el motor de base de datos puede actualizar las estructuras de índice cuando se elimina un objeto del conjunto de datos. La información de la figura debería interpretarse como sigue:

• el lado superior izquierdo de la figura muestra el conjunto de datos original
• el lado superior derecho de la figura muestra el conjunto de datos con el objeto que se eliminará marcado con una línea continua

• el lado inferior derecho de la figura muestra el conjunto de datos con el objeto eliminado y las estructuras de índice actualizadas
• todos los vectores de índice que han sido modificados están en cursiva, todos los demás vectores de índice no han sido modificados

[0083] En este ejemplo, un objeto, identificado por el número 6, se elimina del conjunto de datos. Para mantener el índice de estructuras válido, el motor de base de datos debe actualizar el índice de relación, el índice de identidad de tipo de objeto y el vector de bits de identidad eliminado. El motor de base de datos comienza eliminando el vector de índice de identidad, del objeto eliminado, del vector de índice de identidad de tipo de objeto. En este caso, el vector de índice de identidad, 0000010, se elimina del vector de índice de identidad de tipo de objeto para el tipo de objetos marcado como D, lo que tiene como resultado un vector nuevo de 0010000. La etapa siguiente, para el motor de base de datos, consiste en actualizar los vectores de índice de relación de todos los objetos que relacionados con el objeto eliminado. En este caso, el vector de índice de relación del objeto identificado por el número 1 se actualiza a 1111101, el vector de índice de relación del objeto identificado por el número 4 se actualiza a 1000101 y el vector de índice de relación del objeto identificado por el número 7 se actualiza a 1000101.

[0084] La fig. 16 muestra una representación esquemática de cómo el motor de base de datos crea vectores de índice de identidad para los objetos añadidos al conjunto de datos. Seguir este esquema usando el conjunto de datos representado en la fig. 14 produciría las etapas siguientes que crearán el mismo vector de índice de identidad tal y como se muestra en la fig. 14:

• calcular el número total de objetos, después de que los objetos nuevos se hayan añadido, en el conjunto de datos, en este caso 8 objetos
• crear un contador de objetos y establecer su valor en el número total de objetos del conjunto de datos antes de que se añadieran los objetos nuevos, en este caso 7
• empezar por el primer objeto que se ha añadido, en este caso el objeto identificado por el número 8

    ◦ crear un vector de índice de identidad de longitud igual al mayor del número total de objetos en el conjunto de datos y el vector de índice de identidad más largo, en este caso 8
    ◦ dado que no se ha eliminado ningún objeto en este conjunto de datos, el vector de bits de identidad eliminado no contiene ningún uno y debería crearse un vector de índice de identidad nuevo usando el contador de objetos
    ◦ establecer todos los bits en cero a excepción del bit de mismo número que el contador de objetos, en este caso el número de bit 7
    ◦ el vector de bits resultante es 00000001
    ◦ incrementar en uno el contador de objetos, en este caso el valor nuevo es 8

• dado que no se han añadido más objetos, el proceso está completo

[0085] La fig. 17 muestra una representación esquemática de cómo el motor de base de datos crea vectores de índice de relación para los objetos añadidos al conjunto de datos y actualiza los vectores de índice de relación para los objetos relacionados con los objetos nuevos. Seguir este esquema usando el conjunto de datos representado en la fig. 14 produciría las etapas siguientes:

• empezar por el primer objeto que se ha añadido, en este caso el objeto identificado por el número 8

    ◦ crear un vector de índice de relación igual al vector de índice de identidad de objetos, en este caso 00000001
    ◦ usar operaciones OR a nivel de bits para añadir los vectores de índice de identidad para todos los padres y todos los hijos, en este caso 0001000 y 1000000
    ◦ usar operaciones OR a nivel de bits para añadir la identidad de todos los hermanos de diferentes tipos de objetos y todos los hermanos de diferentes clases de objetos para todos los padres, en este caso ninguno
    ◦ el vector de bits resultante es 10010001
    ◦ crear un vector de bits de identidad de padre igual al vector de índice de identidad de objetos, en este caso 00000001
    ◦ actualizar los vectores de índice de relación para todos los objetos relacionados con este objeto, a excepción de los hijos
    ◦ empezar por el padre del nuevo objeto, en este caso el objeto identificado por el número 3

        ▪ usar una operación OR a nivel de bits para añadir el vector de índice de relación del nuevo objeto al objeto actual, en este caso usar una operación OR a nivel de bit para añadir 10010001 a 1001000 dando como resultado 10010001
        ▪ añadir el vector de índice de identidad del objeto actual al vector de bits de identidad de padre

            • el valor actual del vector de bits de identidad de padre es 00000001

• el vector de índice de identidad del objeto actual es 0001000

• el resultado es 00010001

◦ continuar con el siguiente padre, en este caso el objeto identificado por el número 1

▪ usar una operación OR a nivel de bits para añadir el vector de índice de relación del nuevo objeto al objeto actual, en este caso usar una operación OR a nivel de bits para añadir 10010001 a 1111111 dando como resultado 11111111
▪ añadir el vector de índice de identidad del objeto actual al vector de bits de identidad de padre

• el valor actual del vector de bits de identidad de padre es 00010001

• el vector de índice de identidad del objeto actual es 1000000

• el resultado es 10010001

◦ dado que no existen más padres, esta parte del proceso está completa
◦ actualizar los vectores de índice de relación para todos los hijos de este objeto
◦ dado que no existen hijos, esta parte del proceso está completa

• dado que no se han añadido más objetos, el proceso está completo

[0086] La fig. 18 muestra una representación esquemática de cómo el motor de base de datos actualiza los vectores de índice de relación para los objetos relacionados con los objetos que se han eliminado del conjunto de datos. Seguir este esquema usando el conjunto de datos representado en la fig. 15 produciría las etapas siguientes y crearía los mismos vectores de índice de relación tal y como se muestra en la fig. 15:

• empezar por el primer objeto que se ha eliminado, en este caso el objeto identificado por el número 6

◦ añadir este vector de identidad de objetos al vector de bits de identidad eliminado
◦ crear un vector de bits nuevo llevando a cabo una operación NOT a nivel de bits en el vector de índice de identidad del objeto

▪ el vector de índice de identidad es 0000010
▪ el resultado es 1111101

◦ crear un vector de bits de identidad de padre igual al vector de índice de identidad del objeto, en este caso 0000010
◦ actualizar los vectores de índice de relación para todos los objetos relacionados con este objeto, excepto los hijos
◦ empezar por el padre del objeto eliminado, en este caso el objeto identificado por el número 4

▪ usar una operación AND a nivel de bits, entre el vector de índice de identidad negado del objeto eliminado y el vector de índice de relación del objeto actual para eliminar el vector de índice de identidad del objeto eliminado, en este caso usar una operación AND a nivel de bits para eliminar 0000010 de 1000111 dando como resultado 1000101
▪ añadir el vector de índice de identidad del objeto actual al vector de bits de identidad padre

• el valor actual del vector de bits de identidad de padre es 0000010

• el vector de índice de identidad del objeto actual es 0000100

• el resultado es 0000110

◦ continuar con el primer hermano del objeto eliminado, en este caso el objeto identificado por el número 7

▪ usar una operación AND a nivel de bits, entre el vector de índice de identidad negado del objeto eliminado y el vector de índice de relación del objeto actual para eliminar el vector de índice de identidad del objeto eliminado, en este caso usar una operación AND a nivel de bits para eliminar 0000010 de 1000111 dando como resultado 1000101
▪ añadir el vector de índice de identidad del objeto actual al vector de bits de identidad de padre

• el valor actual del vector de bits de identidad de padre es 0000110

- el vector de índice de identidad del objeto actual es 0000001
- el resultado es 0000111

◦ continuar con el padre siguiente, en este caso el objeto identificado por el número 1

▪ usar una operación AND a nivel de bits, entre el vector de índice de identidad negado del objeto eliminado y el vector de índice de relación del objeto actual para eliminar el vector de índice de identidad del objeto eliminado, en este caso usar una operación AND a nivel de bits para eliminar 0000010 de 1111111 dando como resultado 1111101
▪ añadir el vector de índice de identidad del objeto actual al vector de bits de identidad de padre

- el valor actual del vector de bits de identidad de padre es 0000111
- el vector de índice de identidad del objeto actual es 1000000
- el resultado es 1000111

◦ dado que no existen más padres, esta parte del proceso está completa
◦ actualizar los vectores de índice de relación para todos los hijos de este objeto
◦ dado que no existen hijos, esta parte del proceso es completa

- dado que no se han eliminado más objetos, el proceso está completo

[0087] El vector de bits de identidad eliminado mencionado previamente y usado en el esquema representado en las figuras 16, 17 y 18 se utiliza para permitir que se reutilicen bits, en las estructuras de índice diferentes, previamente usados por los objetos eliminados. Esto permite que el motor de base de datos nunca use vectores de índice de una longitud superior a la cantidad máxima de objetos que han estado presentes en el conjunto de datos.

[0088] La última categoría de actualizaciones del conjunto de datos, donde el motor de base de datos debe ser capaz de actualizar las estructuras de índice, consiste en cambios en las relaciones entre objetos. Si se cambia una relación para un objeto, por ejemplo se cambia el objetivo de una relación o se elimina una relación, las estructuras de índice se actualizan como si el objeto se hubiera eliminado en primer lugar y, a continuación, añadido con la nueva información de relación. De esta manera, el motor de base de datos puede mantener todas las estructuras de índice válidas incluso después de que se haya cambiado una relación.

[0089] Dado que actualizar las estructuras de índice de la manera previamente descrita hará que las estructuras de índices se fragmenten a lo largo del tiempo, existe una necesidad de reconstituir las estructuras de índice después de que se hayan llevado a cabo suficientes actualizaciones. El que las estructuras de índice estén fragmentadas significa, en este contexto, que pueden contener bits sin usar en el lugar en el que estaba un objeto eliminado o que el orden de los objetos en los vectores de bits ya no es óptimo y, por lo tanto, se puede conseguir una tasa de compresión y una velocidad de procesamiento inferiores.

[0090] Aunque se han descrito particularmente determinadas formas de realización ilustrativas de la invención, se entenderá que varias otras modificaciones resultarán fácilmente comprensibles para los expertos en la técnica sin apartarse del ámbito de la invención. Por consiguiente, el alcance de las reivindicaciones aquí anexas no pretende limitarse a la descripción expuesta en la presente, sino que las reivindicaciones deben interpretarse de forma que abarquen todos los equivalentes de la presente invención que resulten evidentes para los expertos en la técnica a la que pertenece la invención.

REIVINDICACIONES

1. Método para proporcionar una estructura de índice en una base de datos que comprende una pluralidad de tipos de objetos, donde cada tipo de objetos comprende al menos un objeto, que incluye las etapas de
establecer una posición de un vector de bits de índice de identidad de cada objeto en dicha base de datos en uno y todas las demás posiciones en cero, donde dicha posición es diferente para cada objeto,
formar, para cada objeto en dicha base de datos, un vector de bits de índice de relación combinando todos los vectores de bits de índice de identidad de objetos con los que dicho cada objeto está relacionado, donde cada vector de bits tiene una longitud igual al número de objetos,
formar un índice de identidad de tipo de objeto que comprende un vector de bits de índice para cada tipo de objetos combinando los vectores de bits de índice de identidad de todos los objetos de cada tipo de objetos y
obtener un subconjunto de datos a partir de una base de datos que contiene un conjunto de datos completo, mediante

la especificación de un primer subconjunto de datos que comprende objetos seleccionados de dicho conjunto de datos completo en la base de datos,
la identificación de todos los objetos de dicho conjunto de datos completo que están incluidos en o relacionados con dicho primer subconjunto de datos y que cumplen un criterio seleccionado, y
la creación de un segundo subconjunto de datos que contiene todos los objetos identificados y todos los objetos relacionados con dichos objetos identificados.

2. Método tal y como se reivindica en la reivindicación 1, que comprende además la etapa de
alinear cada vector de bits de índice a la longitud de palabra de una arquitectura de procesador destinada estableciendo el número de bits en el múltiplo más bajo de la longitud de palabra que puede representar todos los objetos y estableciendo los bits adicionales en cero.

3. Método tal y como se reivindica en la reivindicación 1 o en la reivindicación 2, que comprende además la etapa de separar un conjunto de datos en una pluralidad de particiones, y repetir para cada partición las etapas de:

establecer una posición de un vector de bits de índice de identidad de cada objeto en dicha base de datos en uno y todas las demás posiciones en cero, donde dicha posición es diferente para cada objeto, y
formar, para cada objeto en dicha base de datos, un vector de bits de índice de relación combinando todos los vectores de bits de índice de identidad de objetos con los que dicho cada objeto está relacionado, donde cada vector de bits tiene una longitud igual al número de objetos.

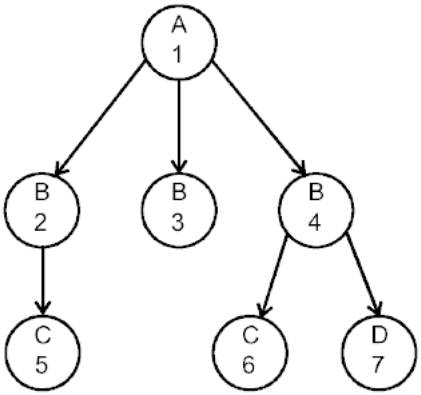
A
1
B
2
B
3
B
4
C
5
C
6
D
7

Fig. 1

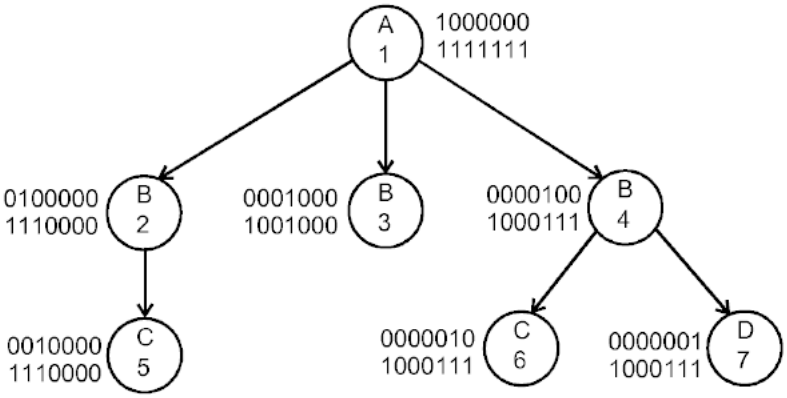
A
1
1000000
1111111
0100000
1110000
B
2
0001000
1001000
B
3
0000100
1000111
B
4
0010000
1110000
C
5
0000010
1000111
C
6
0000001
1000111
D
7

Fig. 3

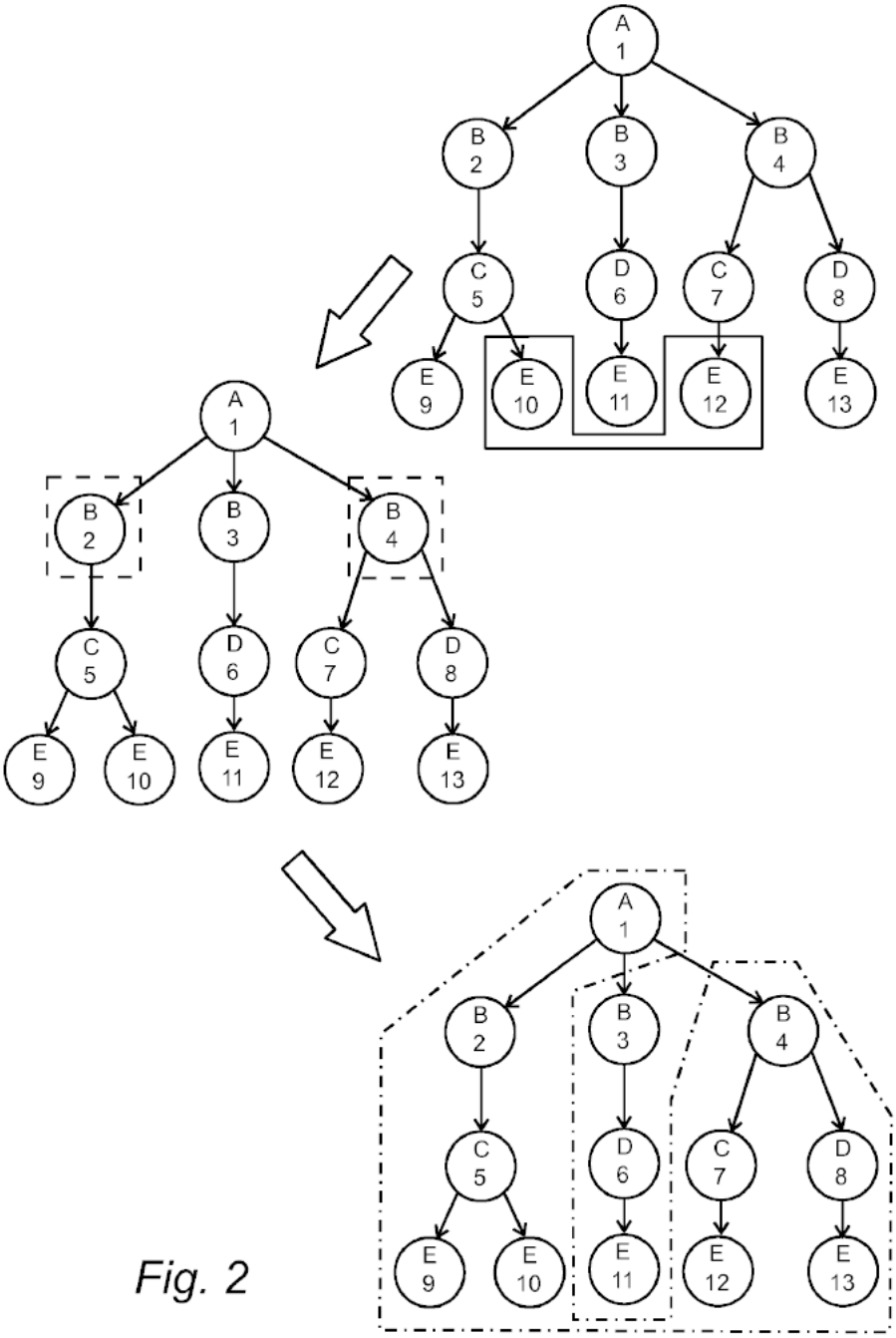
A 1
B 2
B 3
B 4
C 5
D 6
C 7
D 8
E 9
E 10
E 11
E 12
E 13
A 1
B 2
B 3
B 4
C 5
D 6
C 7
D 8
E 9
E 10
E 11
E 12
E 13
A 1
B 2
B 3
B 4
C 5
D 6
C 7
D 8
E 9
E 10
E 11
E 12
E 13

Fig. 2

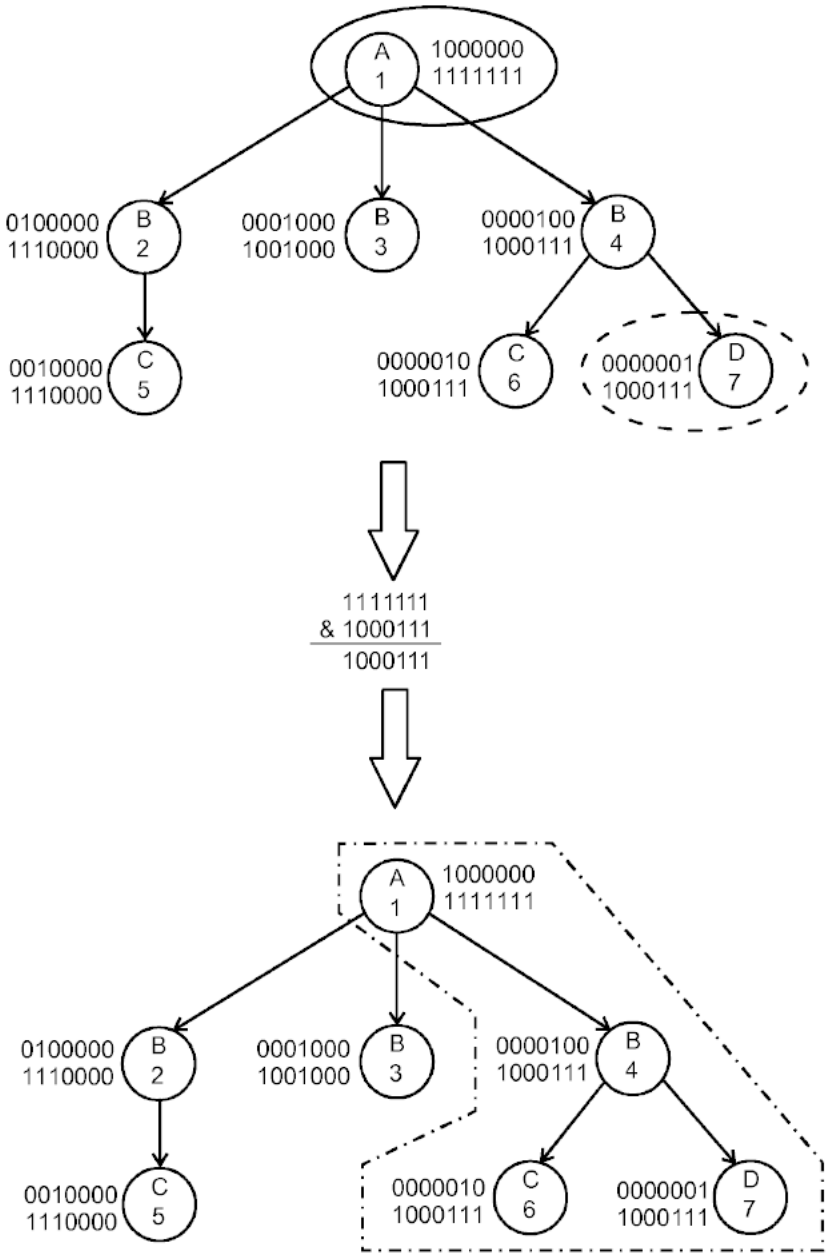
A
1
1000000
1111111
0100000
1110000
B
2
0001000
1001000
B
3
0000100
1000111
B
4
0010000
1110000
C
5
0000010
1000111
C
6
0000001
1000111
D
7
1111111
& 1000111
1000111
A
1
1000000
1111111
0100000
1110000
B
2
0001000
1001000
B
3
0000100
1000111
B
4
0010000
1110000
C
5
0000010
1000111
C
6
0000001
1000111
D
7

Fig. 4

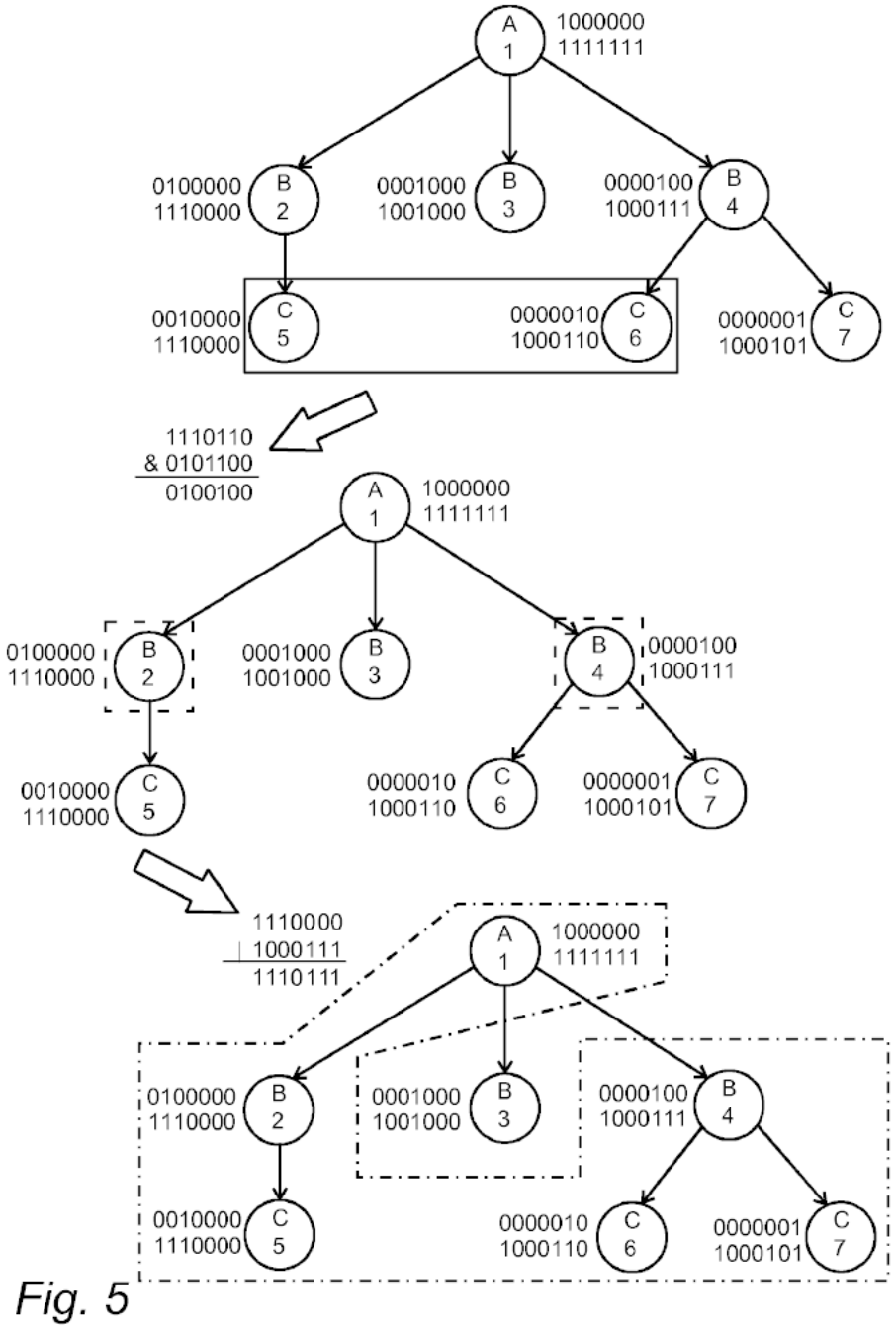
A
1
1000000
1111111
0100000
1110000
B
2
0001000
1001000
B
3
0000100
1000111
B
4
0010000
1110000
C
5
0000010
1000110
C
6
0000001
1000101
C
7
1110110
& 0101100
0100100
A
1
1000000
1111111
0100000
1110000
B
2
0001000
1001000
B
3
B
4
0000100
1000111
0010000
1110000
C
5
0000010
1000110
C
6
0000001
1000101
C
7
1110000
& 1000111
1110111
A
1
1000000
1111111
0100000
1110000
B
2
0001000
1001000
B
3
0000100
1000111
B
4
0010000
1110000
C
5
0000010
1000110
C
6
0000001
1000101
C
7

Fig. 5

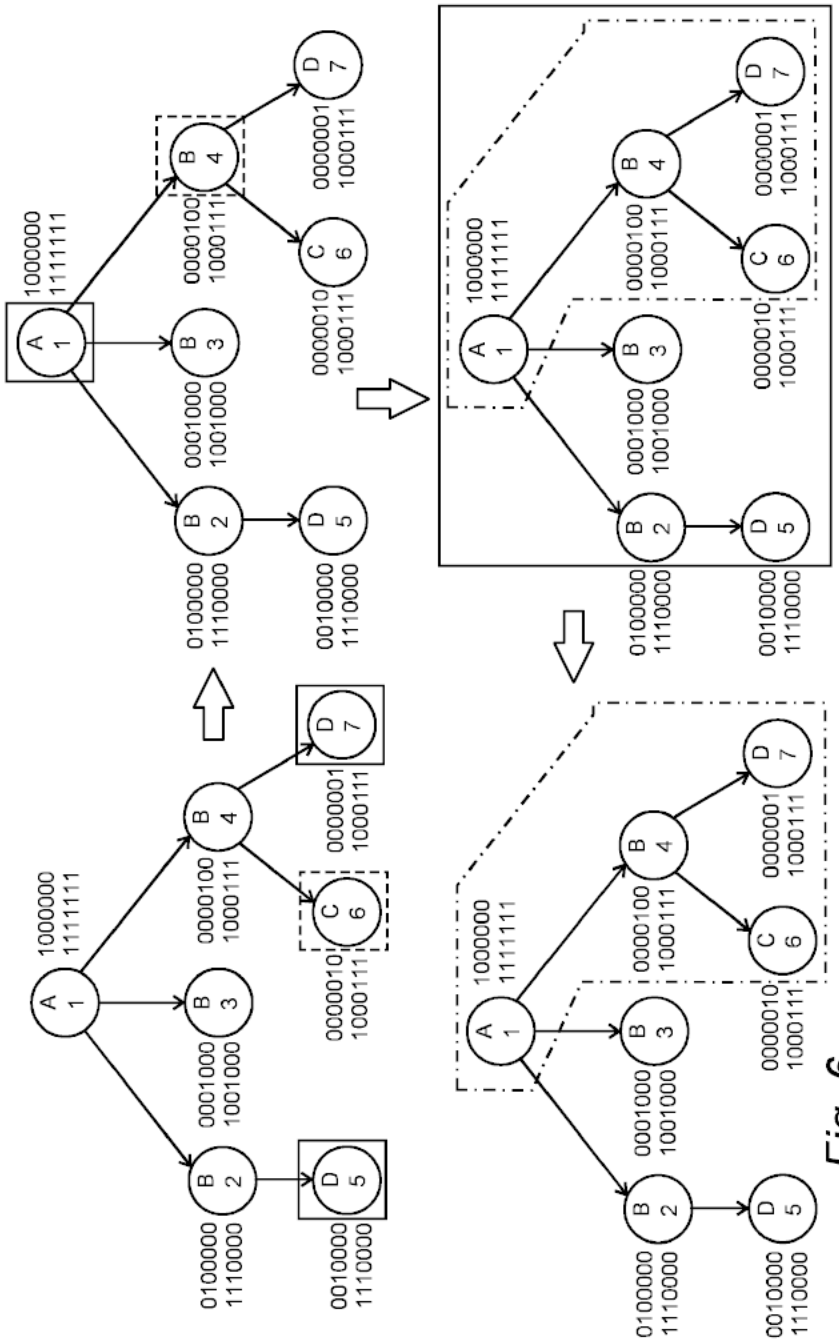
A 1
1000000 1111111
B 2
0100000 1110000
B 3
0001000 1001000
B 4
0000100 1000111
C 6
0000010 1000111
D 5
0010000 1110000
D 7
0000001 1000111

Fig. 6

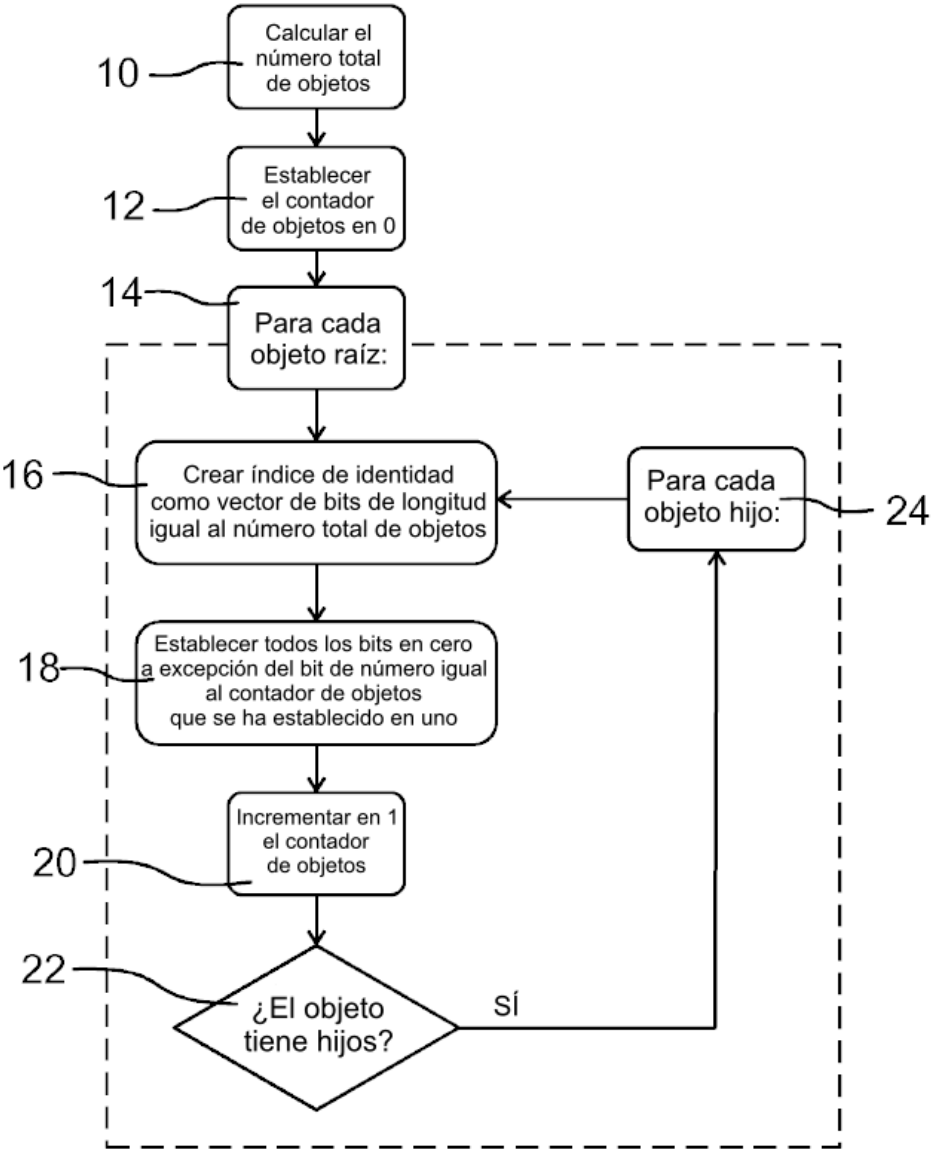
Calcular el número total de objetos
10
Establecer el contador de objetos en 0
12
Para cada objeto raíz:
14
Crear índice de identidad como vector de bits de longitud igual al número total de objetos
16
Para cada objeto hijo:
24
Establecer todos los bits en cero a excepción del bit de número igual al contador de objetos que se ha establecido en uno
18
Incrementar en 1 el contador de objetos
20
¿El objeto tiene hijos?
22
SÍ

Fig. 7

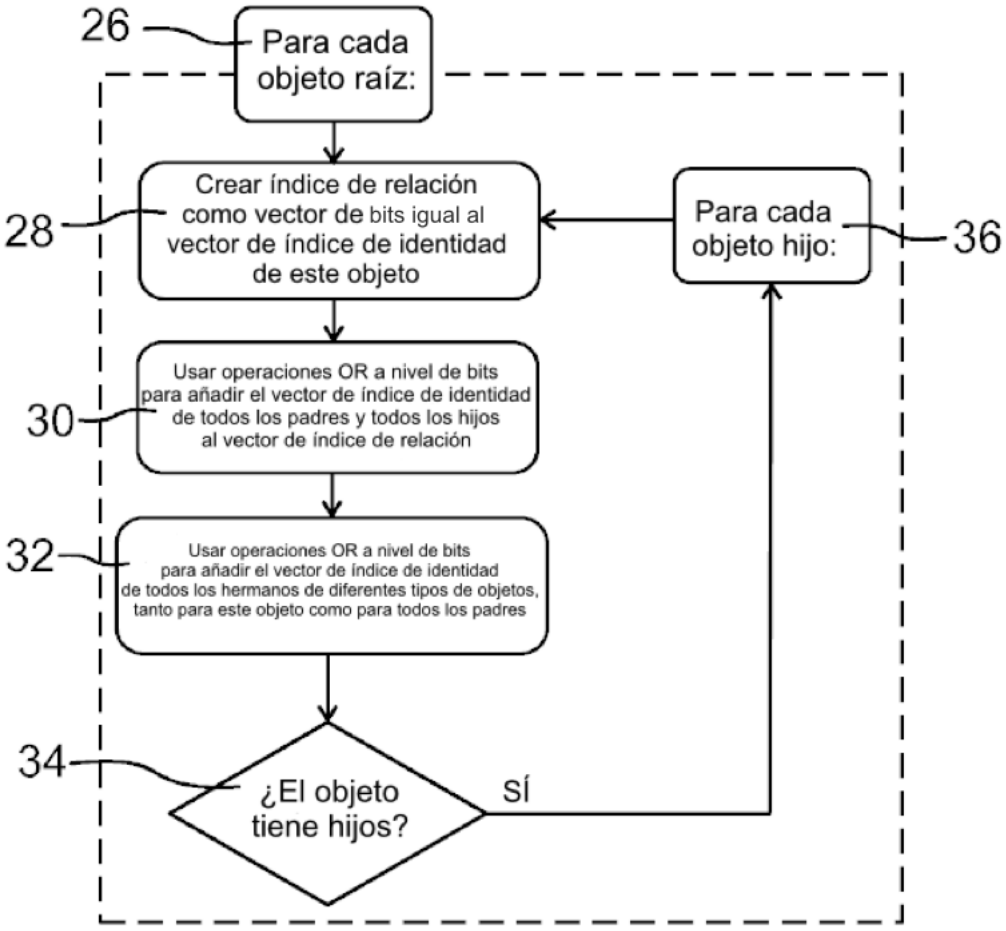
26
Para cada objeto raíz:
28
Crear índice de relación como vector de bits igual al vector de índice de identidad de este objeto
Para cada objeto hijo:
36
30
Usar operaciones OR a nivel de bits para añadir el vector de índice de identidad de todos los padres y todos los hijos al vector de índice de relación
32
Usar operaciones OR a nivel de bits para añadir el vector de índice de identidad de todos los hermanos de diferentes tipos de objetos, tanto para este objeto como para todos los padres
34
¿El objeto tiene hijos?
SÍ

Fig. 8

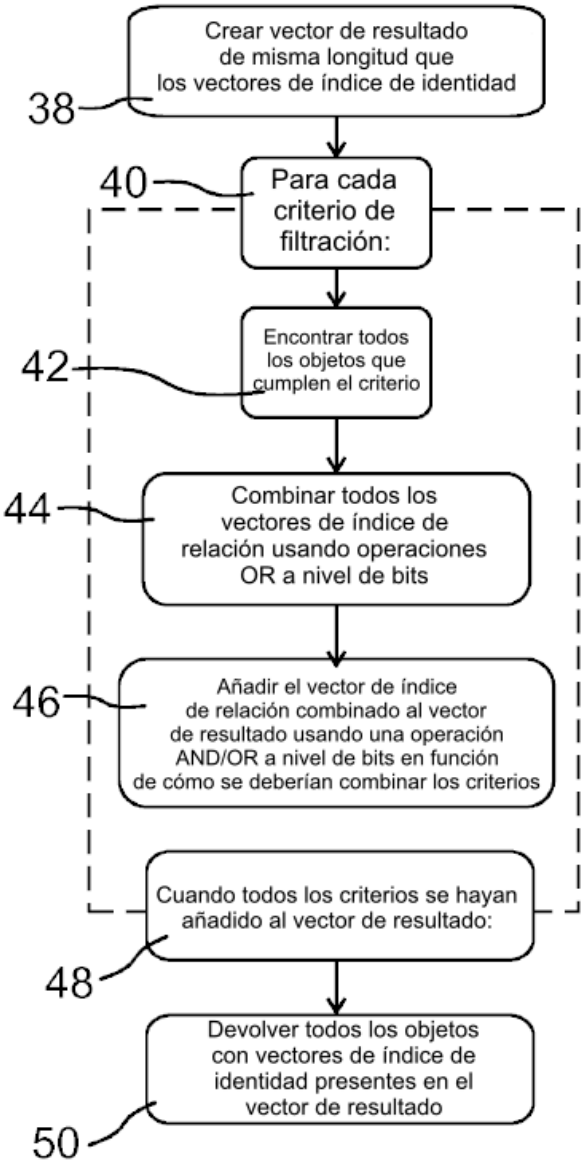
Crear vector de resultado de misma longitud que los vectores de índice de identidad
38
40
Para cada criterio de filtración:
Encontrar todos los objetos que cumplen el criterio
42
44
Combinar todos los vectores de índice de relación usando operaciones OR a nivel de bits
46
Añadir el vector de índice de relación combinado al vector de resultado usando una operación AND/OR a nivel de bits en función de cómo se deberían combinar los criterios
Cuando todos los criterios se hayan añadido al vector de resultado:
48
Devolver todos los objetos con vectores de índice de identidad presentes en el vector de resultado
50

Fig. 9

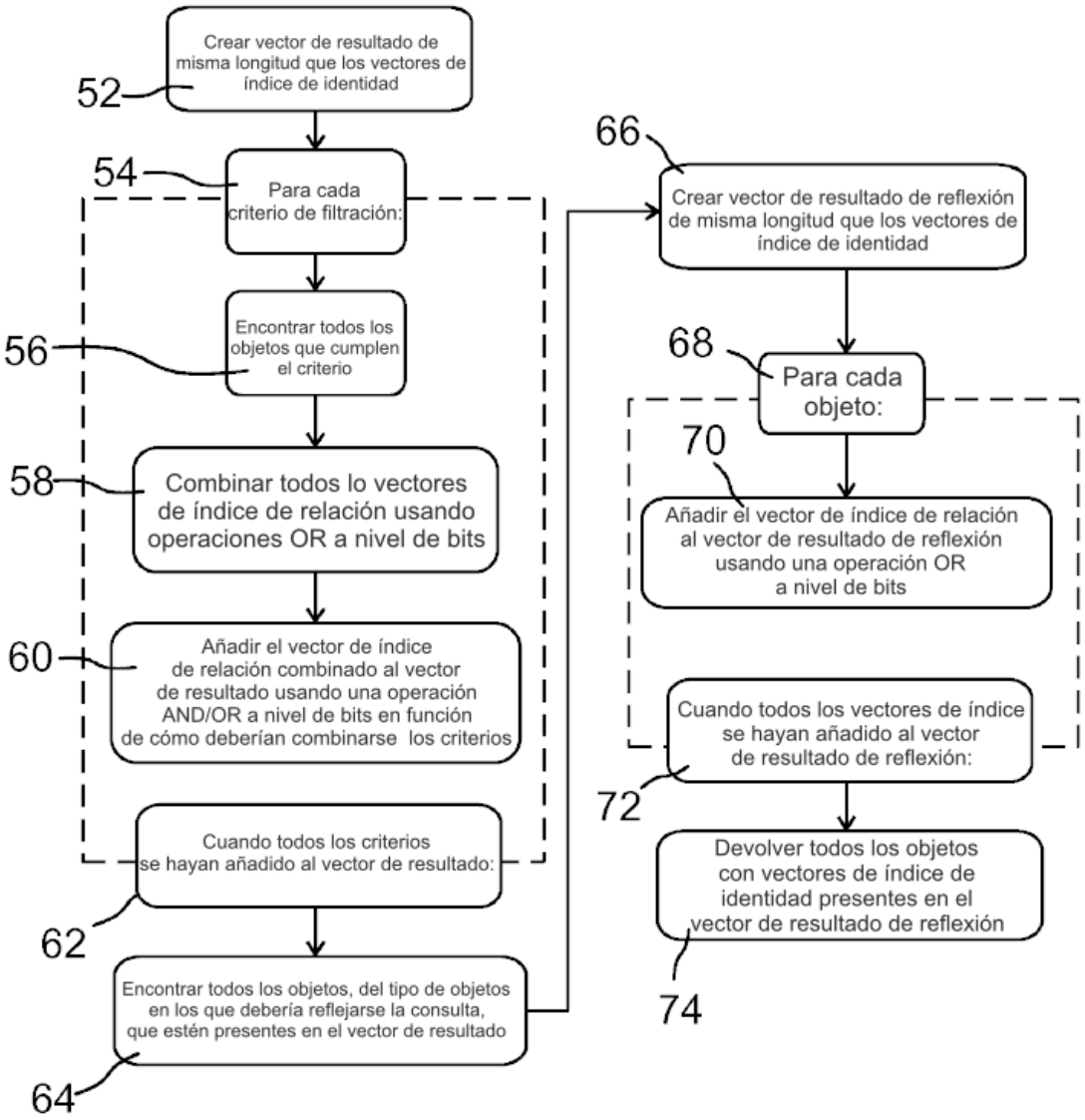
Crear vector de resultado de misma longitud que los vectores de índice de identidad
52
54
Para cada criterio de filtración:
Encontrar todos los objetos que cumplen el criterio
56
58
Combinar todos lo vectores de índice de relación usando operaciones OR a nivel de bits
60
Añadir el vector de índice de relación combinado al vector de resultado usando una operación AND/OR a nivel de bits en función de cómo deberían combinarse los criterios
Cuando todos los criterios se hayan añadido al vector de resultado:
62
Encontrar todos los objetos, del tipo de objetos en los que debería reflejarse la consulta, que estén presentes en el vector de resultado
64
66
Crear vector de resultado de reflexión de misma longitud que los vectores de índice de identidad
68
Para cada objeto:
70
Añadir el vector de índice de relación al vector de resultado de reflexión usando una operación OR a nivel de bits
Cuando todos los vectores de índice se hayan añadido al vector de resultado de reflexión:
72
Devolver todos los objetos con vectores de índice de identidad presentes en el vector de resultado de reflexión
74

Fig. 10

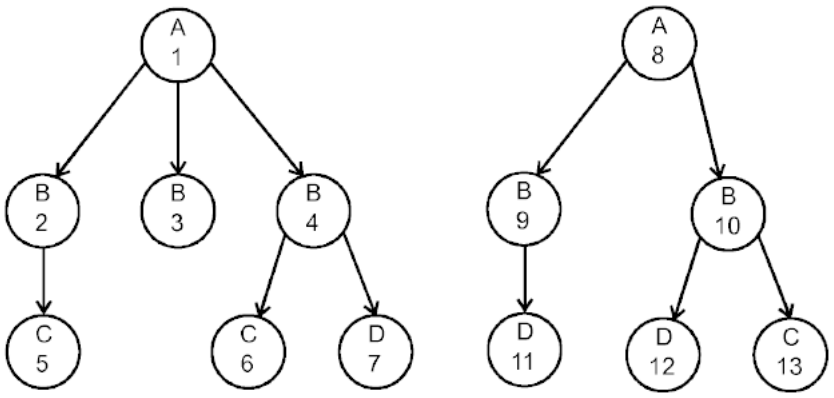
A
1
B
2
B
3
B
4
C
5
C
6
D
7
A
8
B
9
B
10
D
11
D
12
C
13

Fig. 11

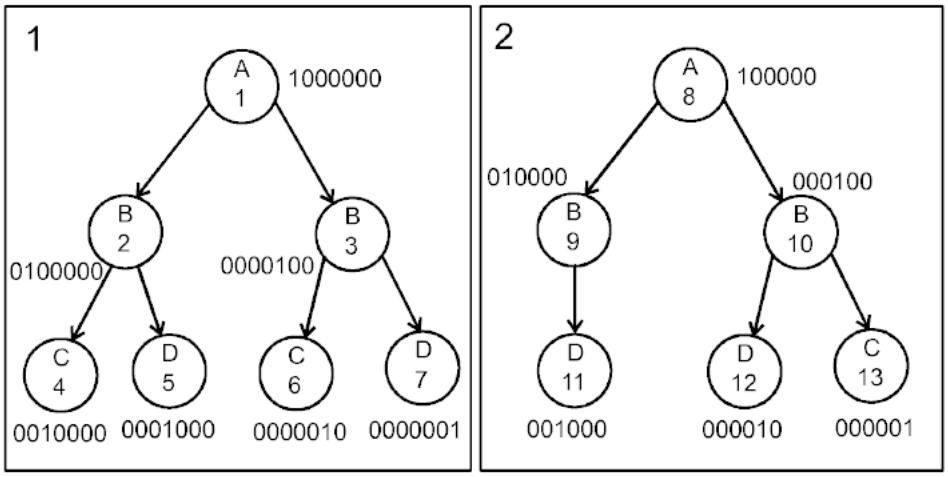
1
A
1
1000000
B
2
B
3
0100000
0000100
C
4
D
5
C
6
D
7
0010000
0001000
0000010
0000001
2
A
8
100000
010000
000100
B
9
B
10
D
11
D
12
C
13
001000
000010
000001

Fig. 12

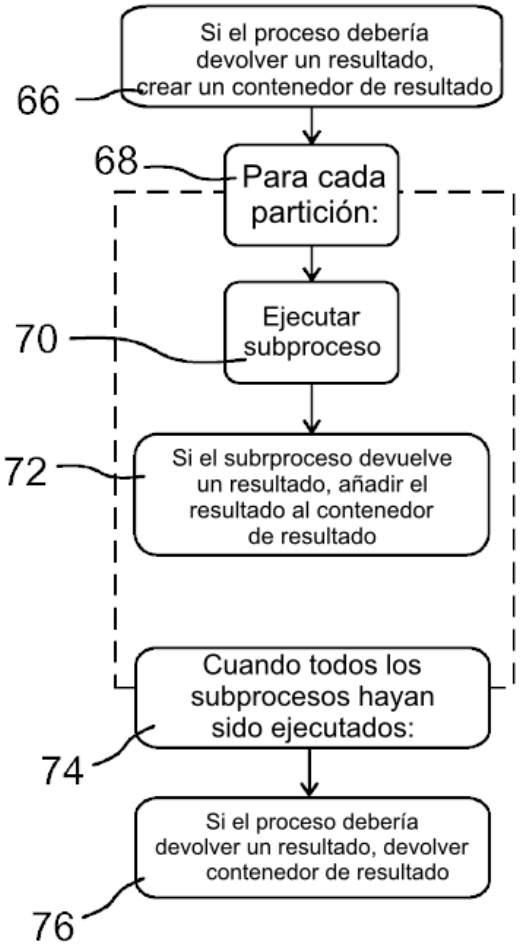
Si el proceso debería devolver un resultado, crear un contenedor de resultado
66
68
Para cada partición:
Ejecutar subproceso
70
Si el subrproceso devuelve un resultado, añadir el resultado al contenedor de resultado
72
Cuando todos los subprocesos hayan sido ejecutados:
74
Si el proceso debería devolver un resultado, devolver contenedor de resultado
76

Fig. 13

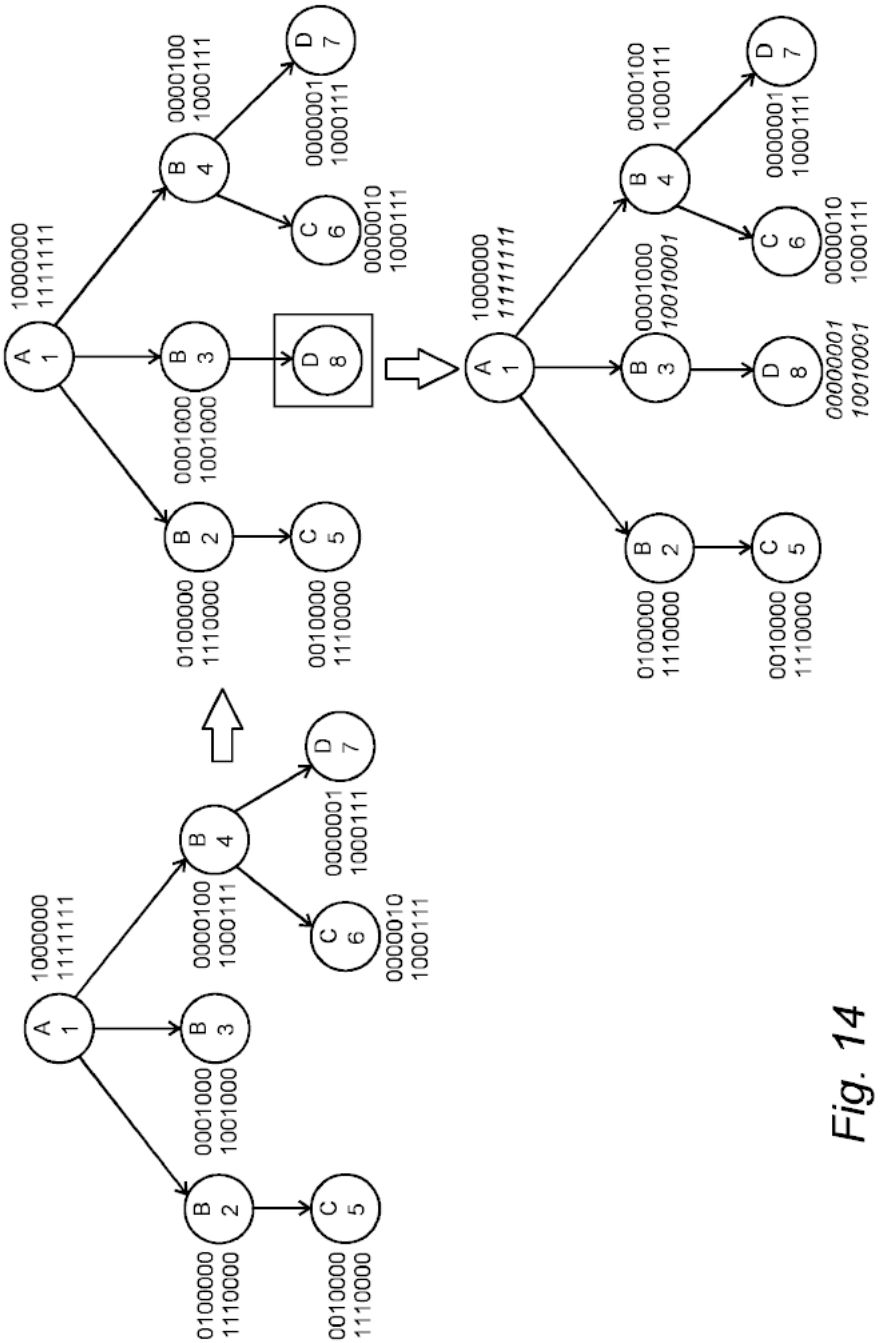
A 1
1000000
11111111
B 2
0100000
1110000
B 3
0001000
1001000
B 4
0000100
1000111
C 5
0010000
1110000
C 6
0000010
1000111
D 7
0000001
1000111
D 8
00000001
10010001

Fig. 14

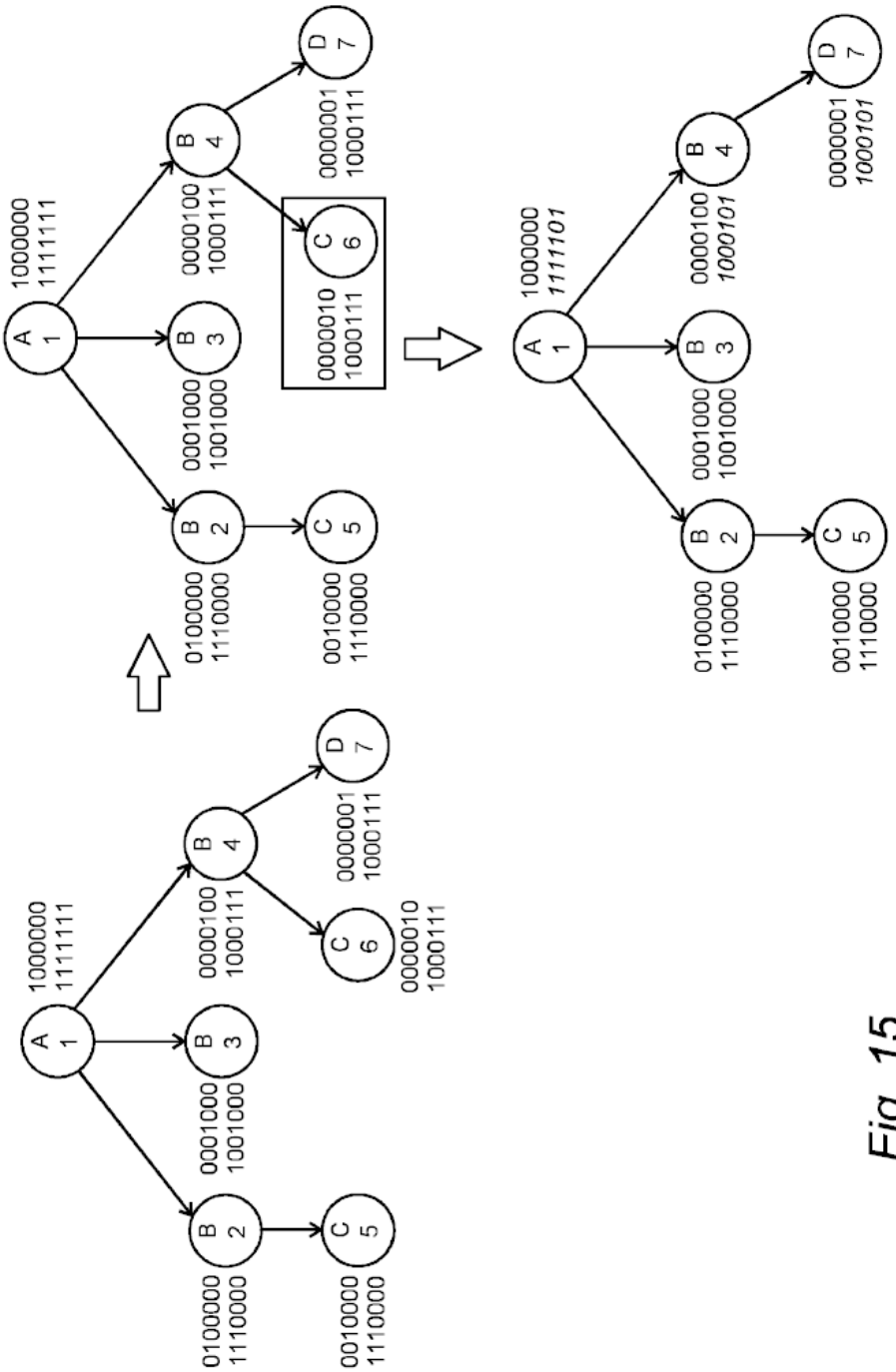
A 1
1000000 1111111
B 2
0100000 1110000
B 3
0001000 1001000
B 4
0000100 1000111
C 5
0010000 1110000
C 6
0000010 1000111
D 7
0000001 1000111
A 1
1000000 1111101
B 2
0100000 1110000
B 3
0001000 1001000
B 4
0000100 1000101
C 5
0010000 1110000
D 7
0000001 1000101

Fig. 15

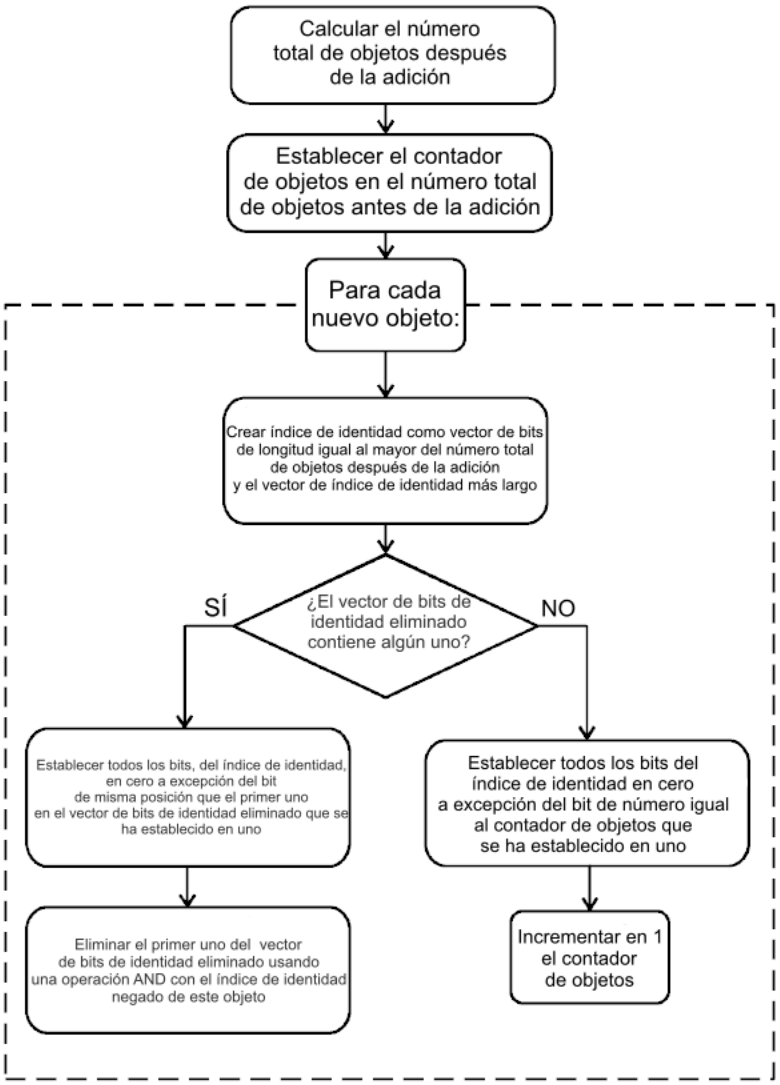
Calcular el número total de objetos después de la adición
Establecer el contador de objetos en el número total de objetos antes de la adición
Para cada nuevo objeto:
Crear índice de identidad como vector de bits de longitud igual al mayor del número total de objetos después de la adición y el vector de índice de identidad más largo
¿El vector de bits de identidad eliminado contiene algún uno?
SÍ
NO
Establecer todos los bits, del índice de identidad, en cero a excepción del bit de misma posición que el primer uno en el vector de bits de identidad eliminado que se ha establecido en uno
Eliminar el primer uno del vector de bits de identidad eliminado usando una operación AND con el índice de identidad negado de este objeto
Establecer todos los bits del índice de identidad en cero a excepción del bit de número igual al contador de objetos que se ha establecido en uno
Incrementar en 1 el contador de objetos

Fig. 16

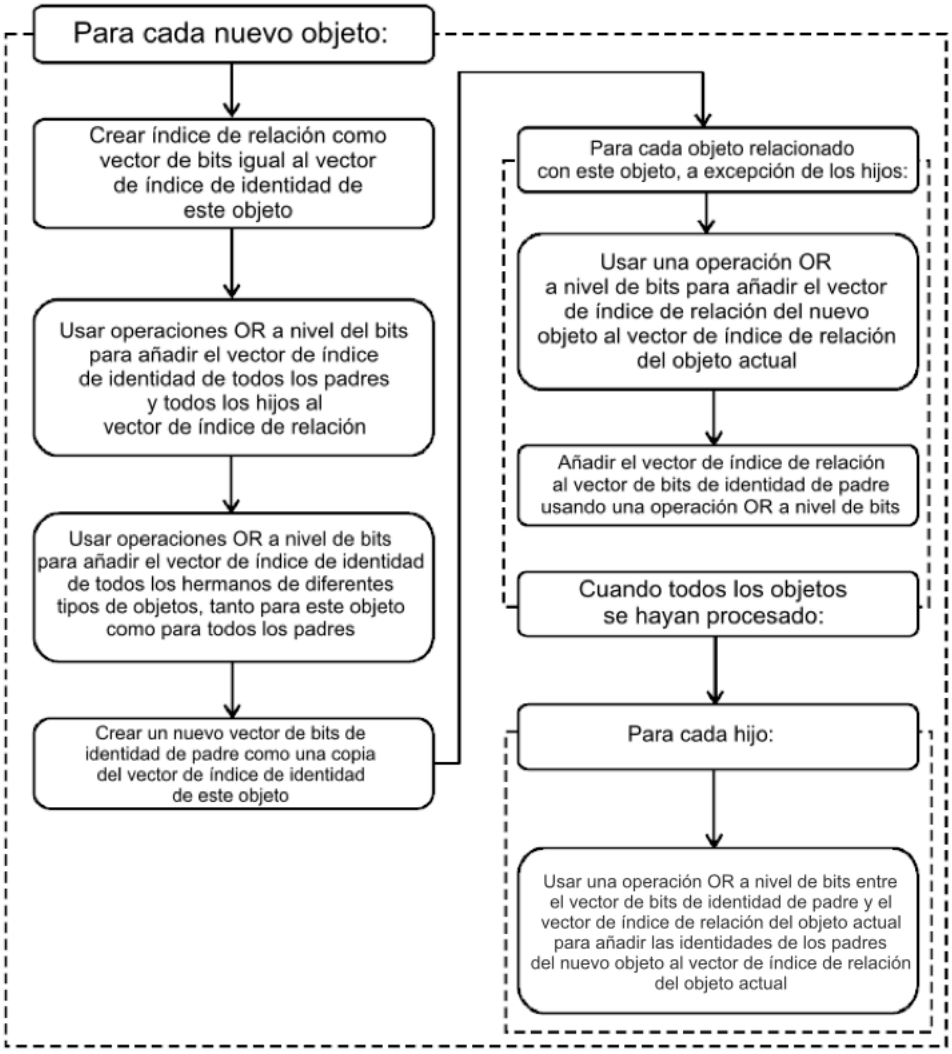
Para cada nuevo objeto:
Crear índice de relación como vector de bits igual al vector de índice de identidad de este objeto
Usar operaciones OR a nivel del bits para añadir el vector de índice de identidad de todos los padres y todos los hijos al vector de índice de relación
Usar operaciones OR a nivel de bits para añadir el vector de índice de identidad de todos los hermanos de diferentes tipos de objetos, tanto para este objeto como para todos los padres
Crear un nuevo vector de bits de identidad de padre como una copia del vector de índice de identidad de este objeto
Para cada objeto relacionado con este objeto, a excepción de los hijos:
Usar una operación OR a nivel de bits para añadir el vector de índice de relación del nuevo objeto al vector de índice de relación del objeto actual
Añadir el vector de índice de relación al vector de bits de identidad de padre usando una operación OR a nivel de bits
Cuando todos los objetos se hayan procesado:
Para cada hijo:
Usar una operación OR a nivel de bits entre el vector de bits de identidad de padre y el vector de índice de relación del objeto actual para añadir las identidades de los padres del nuevo objeto al vector de índice de relación del objeto actual

Fig. 17

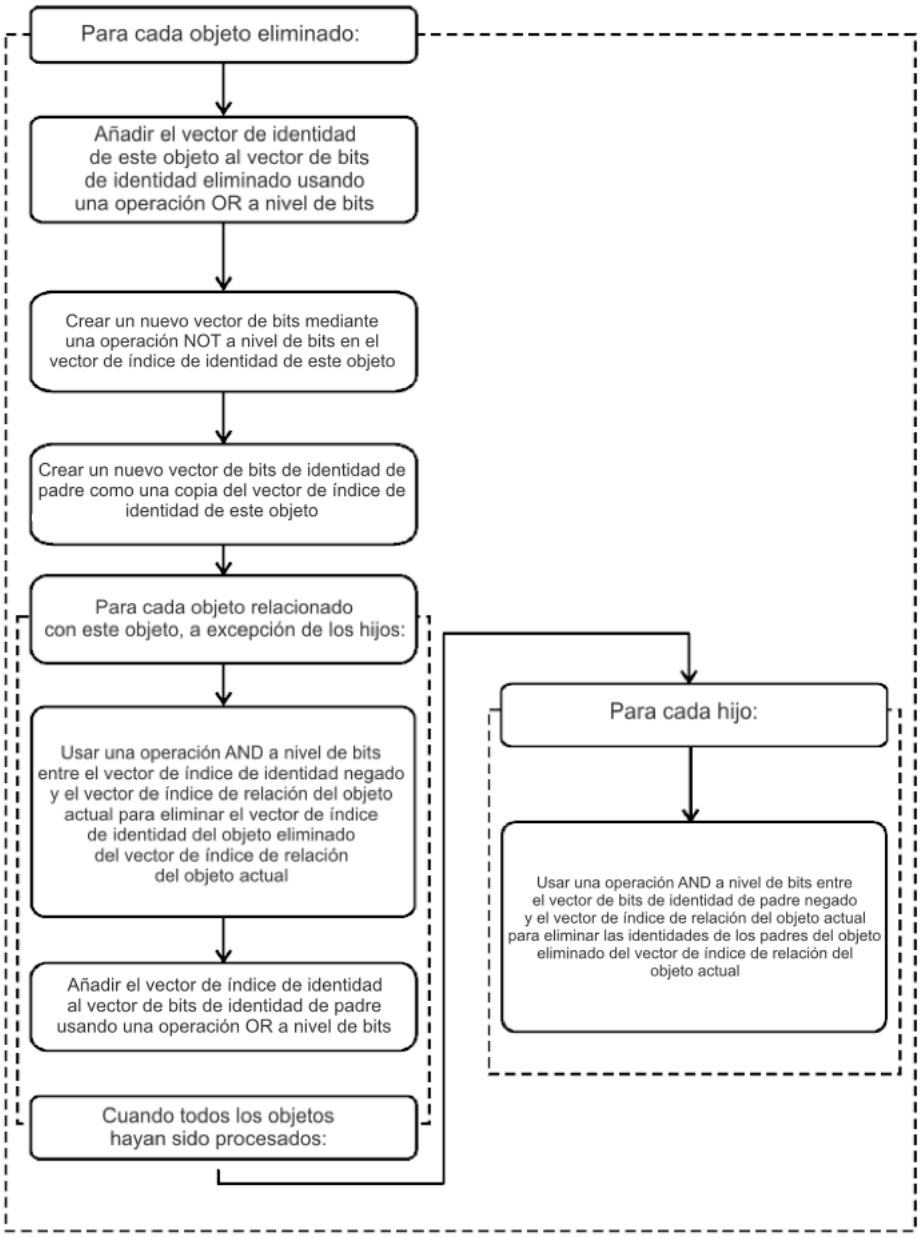
Para cada objeto eliminado:
Añadir el vector de identidad de este objeto al vector de bits de identidad eliminado usando una operación OR a nivel de bits
Crear un nuevo vector de bits mediante una operación NOT a nivel de bits en el vector de índice de identidad de este objeto
Crear un nuevo vector de bits de identidad de padre como una copia del vector de índice de identidad de este objeto
Para cada objeto relacionado con este objeto, a excepción de los hijos:
Usar una operación AND a nivel de bits entre el vector de índice de identidad negado y el vector de índice de relación del objeto actual para eliminar el vector de índice de identidad del objeto eliminado del vector de índice de relación del objeto actual
Añadir el vector de índice de identidad al vector de bits de identidad de padre usando una operación OR a nivel de bits
Cuando todos los objetos hayan sido procesados:
Para cada hijo:
Usar una operación AND a nivel de bits entre el vector de bits de identidad de padre negado y el vector de índice de relación del objeto actual para eliminar las identidades de los padres del objeto eliminado del vector de índice de relación del objeto actual

Fig. 18