UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

CAMPUS ARAGÓN

"SISTEMA Web PARA EL CALCULO DE LA RUTA OPTIMA DENTRO DE UNA ORGANIZACIÓN DE DISTRIBUCIÓN, EMPLEANDO ALGORITMOS DE INTELIGENCIA ARTIFICIAL"

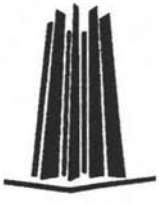
T E S I S

QUE PARA OBTENER EL TÍTULO DE:
INGENIERO EN COMPUTACIÓN

P R E S E N T A:

JORGE RODRÍGUEZ RUBIO

ASESOR DE TESIS:
ING. VÍCTOR RAMÓN AGUILAR OCAMPO



MÉXICO, 2005.

A mis padres, Venustiano y Mercedes,
que les debo todo lo que soy.

A mis hermanos, amigos y compañeros que han estado,
junto a mi, a lo largo de gran parte de la vida.

Agradecimientos.

Señor y Dios mio, el camino de la vida es fácil de recorrer si sigo tus huellas, las vicisitudes desaparecen al estar en tu regazo, y una sola hoja no se mueve sin tu voluntad. Por eso humildemente te doy las gracias, porque en tu infinita bondad haz permitido que fructifique y al mismo tiempo me haz concedido culminar esta meta en el camino de la superación profesional.

Gracias a mi padre que siempre estuvo junto a mí incondicionalmente ademas de brindarme siempre su amor, a mi madre quien siempre ha sido mi apoyo incondicional brindandome su infinito amor de forma desinteresada.

Mi agradecimiento a cuantas personas han hecho posible la realización del presente trabajo con cita especial del Ing. Victor Ramon Aguilar quien me tuvo paciencia y gracias a sus consejos fue posible la creación del presente. A mis revisores: el Ing. César Francisco Germán, el Ing. Rodolfo Vázquez Morales, el MTI. Omar Mendoza González y el M. en C. Marcelo Pérez quien no solo es una increíble persona sino tambien comparte lo mas valioso que cualquier persona puede tener: su maravilloso conocimiento y experiencia, a todos los profesores de nuestra querida alma matter que durante la mitad de mi vida academica no solo me han formado sino me han transmitido toda su experiencia y conocimiento.

A todos mis hermanos con los que siempre he contado y me han brindado su cariño y comprensión, a todos mis amigos que es el patrimonio mas importante que alguien puede poseer en especial me siento extremadamente afortunado por contar con la amistad del Ing. David Guerrero Guerra que no solo ha sido una excelsa compañía a lo largo de mi trayectoria academica sino me ha brindado su maravillosa amistad que durante tantos años solo se ha forjado en lo mas profundo de mi ser , no existen las palabras para describir lo valioso que es una verdadera amistad ¡GRACIAS!.

Gracias a todos mis amigos-compañeros de nuestra alma matter que no solo pase el tiempo mas maravilloso de mi vida , sino que comparti muchos años junto a ellos, a mi novia por estar conmigo y brindarme su preciosa compañía durante gran parte de mi vida.

Muchas gracias a todos mis sobrinos que no solo me hacen extremadamente feliz sino que disfruto plenamente su compañía dia tras dia.

Gracias a dios por permitirme vivir y disfrutar de lo bello que es VIVIR, ya que el secreto de la existencia no consiste en solo vivir sino en saber para que se vive.

Un amigo es uno que lo sabe todo de ti y a pesar de ello te quiere.

Índice de Contenido.

Índice de Tablas.

Índice de Ilustraciones.

INTRODUCCIÓN.

Actualmente en México existen una cantidad impresionante de pequeñas y micro empresas, principalmente de origen Mexicano, que tienen un potencial de crecimiento enorme, desafortunadamente la gran mayoría carece de soporte de sistemas informáticos en el área administrativa para la automatización de operaciones y procesos debido a el alto costo que representa adquirir licencias de software comercial así como la inexistencia de la adecuación de dicho software por ser software registrado, que raramente resulta ideal al 100 % tal cual para las reglas del negocio, esto trae consigo un impedimento para el crecimiento de las mismas.

Así al no existir posibilidades de crecimiento existen dos opciones de mercado, marginar a la empresa a sólo llevar a cabo las actividades para subsistir sin tener perspectivas de crecimiento , o ser desplazada por grandes consorcios empresariales en un 95 % de origen extranjero, lo que trae consigo un sin fin de problemas de carácter económico y social para el país , que en su mayoría son desempleo , poco capital de origen nacional , poco crecimiento económico, zonas muy marcadas de poco desarrollo económico , gran cantidad de horas hombre empleadas en el traslado a la zona laboral etc.

Las empresas que subsisten laboran sobre procesos y metodologías no automatizadas que provoca perdida economica asi como duplicación de funciones lo que conlleva a gastos importantes para la operación como lo es un consumo excesivo de combustible, desgaste de unidades, aumento de consumo de horas de trabajo efectivas, eficacia del servio, etc. Una vez identificada esta problemática aunada a las ya mencionadas se realiza la investigación sobre como optimizar la operación de los servicios que provee la organización, mediante la optimizacion de recorridos o de rutas de distribución del area metropolitana de la ciudad de México esto mediante algoritmos de optimizacion.

Es interesante mencionar que dichos algoritmos se analizan intensamente en el area de inteligencia artificial ya que con ello permite a las maquinas simular la toma de deciones de acuerdo a la base de conocimiento que se tenga. Para esto tambien es necesario preguntar acerca de la relativa dificultad computacional de las funciones computables. Este es el objetivo de la Complejidad Algorítmica.

Rabin curioso investigador en 1960 se planteó una pregunta general: ¿Qué quiere decir que f sea más difícil de computar que g? Rabin ideó una axiomática que fue la base para el desarrollo de la teoría de la complejidad abstracta de Blum y otros autores. En 1965 J. Hartmanis y R. E. Stearns escribieron el artículo "On the computational complexity of algorithms", gracias a este trabajo se le dio nombre a este cuerpo de conocimiento. Se fundamentó así la medida de complejidad definida como el tiempo de computación sobre una máquina de Turing multicinta, y se demostraron los teoremas de jerarquía.

De esta forma tenemos la base para definir que funcion es mas compleja para computar que otra , como se observara en el documento tenemos que es muy importante definir este argumento debido a la naturaleza del problema para

optimizar , ya que si no se plantea adecuadamente podra involucrar una cantidad exagerada de tiempo para su funcionamiento,

Para ello en el presente trabajo se presenta la investigación sobre análisis de algoritmos para optimizar procesos y su implementeacion para un sistema informático, esto orientado para motivar el crecimiento y automatizar procesos y operaciones en una empresa privada de Logística y Distribución, mediante un sistema Web, empleando software libre[1] .

Cuyo objetivo general del proyecto es proporcionar las bases suficientes para que desarrolle a corto plazo una solución satisfactoria a dicha organización, para su administración integral y solución de las operaciones que realiza. Asi como un material de consulta para las diversas aplicaciones que requieran optimizar procesos mediante algoritmos computables.

La presente tesis se distribuye de la siguiente manera:

. Capítulo 1: Se presentan conceptos teóricos importantes para el analisis de sistemas web y su desarrollo asi como el estudio de algoritmos y su eficiencia.

. Capítulo 2: Es importante realizar un analisis exhaustivo de la situación actual en la que se encuentra la organización para poder proponer soluciones .

. Capítulo 3: Una vez hecho el analisis y teniendo el resultado de la investigación de algoritmos se procede a realizar un modelo para su implementacion.

. Capítulo 4: Se procede a realizar el diseño del sistema y sus estrategias para su correcta implementacion en la organización.

.

[1] Software Libre: Software sin registro de autor con distribución y modificación libre.

PRIMER CÁPITULO "CONCEPTOS PRELIMINARES."

1. Conceptos Generales.

Inicialmente para comprender el análisis, diseño, desarrollo e implementación de un sistema debemos de comprender diversos conceptos que generalmente damos por entendidos pero no conocemos el alcance del mismo, uno de ellos es el de sistema, ya que dicha palabra la usamos tan comúnmente en nuestra vida cotidiana aplicada a un sin fin de cosas como por ejemplo:

-El sistema de transporte colectivo metro.
-El sistema nervioso.
-El sistema solar.
-El sistema de educación.

Estos ejemplo son por mencionar algunos, pero aquí surge la pregunta ¿que tienen en común sistemas tan distintos como el sistema solar y el sistema de educación? , o en que se asemeja usando la palabra sistema y ¿Como aplica en nuestra área de estudio como sistemas de Información o sistema Web? Iniciaremos definiendo sistema:

Sistema: "*"Conjunto de cosas que ordenadamente relacionadas entre sí contribuyen a un determinado objetivo*"

Así partiendo de esta definición ya somos capaces de discrepar en todas las diferentes formas que puede aplicarse el mismo. Definiendo más a fondo el de nuestra área de estudio tenemos que:

La Información son los datos procesados de forma que sean de utilidad para el receptor de los mismos y a la vez los datos son la Materia Prima de la Información, estos se generan a partir de hechos, acontecimientos y transacciones.

Un Sistema Informático es conjunto de equipos hardware, unidades de software y procedimientos para el uso de los mismos cuyo objetivo es procesar datos para obtener información de utilidad en un ámbito determinado u organización. Así tenemos finalmente que un Sistema De Información es un conjunto de procedimientos desarrollados para llevar a cabo un determinado proceso u operación.

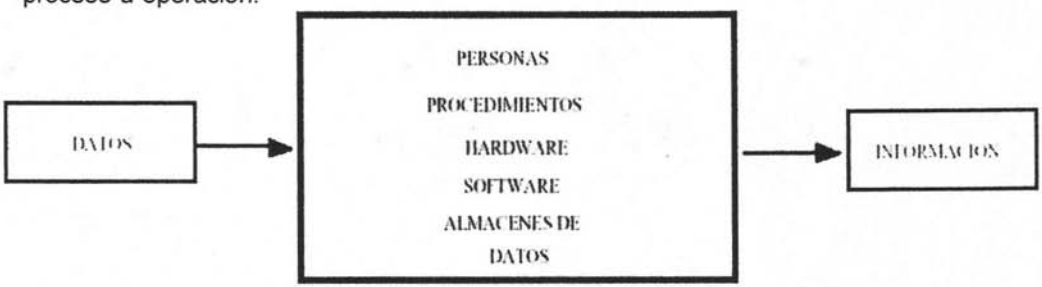


Ilustración 1"Representación de un Sistema"

Cabe señalar que la información debe de cubrir propiedades propias para que sea útil, estas propiedades son: la información debe de ser confiable, debe de estar disponible, debe de ser comprobable y real, a la vez debe de responder a las preguntas ¿Quién? , ¿Cuándo?, ¿Cómo? Y ¿Dónde? Para las diferentes actividades que conllevara el uso del sistema.

Una vez definido "sistema de Información "definiremos brevemente que implica el análisis y diseño de sistemas, así el análisis de sistemas es el proceso de interpretación de los hechos, identificación de los problemas, y empleo de la información para mejorar el funcionamiento de un sistema. En dónde Análisis (Aplicado a Informática.) es el estudio mediante técnicas informáticas de los límites, características, y posibles soluciones de un problema al que se aplica un tratamiento por computadora.
Para esto es necesario:

- **Comprender** como funciona la organización (Se estudia en el Capitulo 2).
- **Interpretar** los hechos y el funcionamiento.
- **Valorar** las futuras repercusiones del la mecanización del sistema en la Organización de la empresa para posibles cambios administrativos y operativos.
- **Determinar** en que forma puede mejorarse los diferentes procesos administrativos.
- **Estudiar** las necesidades futuras de la empresa, teniendo en cuenta posibles modificaciones (que el sistema sea mantenible).
- **Recomendar** la opción u opciones que pueden llevarse a cabo para realizar las mejoras en la organización.
- **Evaluar** Los costos y beneficios que estas mejoras conllevan.

Es de gran importancia destacar estos aspectos para tener un buen sistema informático bien elaborado y más aun ¿Cómo podemos distinguir un sistema bien hecho y de uno que no lo esta? Ya que aquí se remarca la diferencia entre un Analista de Sistemas y un Programador de Sistemas en donde este ultimo realiza el sistema para las tareas que el usuario necesita y un buen sistema NO es aquel que únicamente realiza las tareas que el usuario necesita. Tenemos que observar los siguientes puntos para el desarrollo del sistema:

- **El sistema ha de ser mantenible.** Con esto debemos de considerar la "evolución" de nuestro sistema a posibles modificaciones para que se adapte a un entorno que no es estático.
- **El software debe de ser confiable:** Es muy importante este punto ya que un ERROR EN EL DISEÑO O EN LA PROGRAMACIÓN = ERRORES EN EL FUNCIONAMIENTO DEL SISTEMA. Y actualmente un error no es sólo un reporte incorrecto o una falla mínima, un error es un costo y el costo puede ser de tiempo económico o más aun de VIDAS HUMANAS.
- **El software debe ser eficiente**: La información debe de estar disponible (propiedad de la información).
- **Debe proporcionar una interfaz** apropiada con el usuario. En este punto se debe de poner principal atención ya que el éxito de un sistema depende en gran medida de la aceptación y facilidad para los usuarios, ya que al observar la siguiente tabla podemos ver el enorme fracaso que puede tener un sistema:

No.	Porcentaje	Descripción
1	1.5%	Se uso tal como se entrego.
2	3.5%	Se uso después de algunos cambios.
3	19.0%	Se uso luego se abandono o rechazo.
4	47%	Se entrego pero nunca se uso.
5	29%	Se pago para su desarrollo pero nunca se termino.

Tabla 1 "Porcentaje de la problemática en el desarrollo de sistemas"

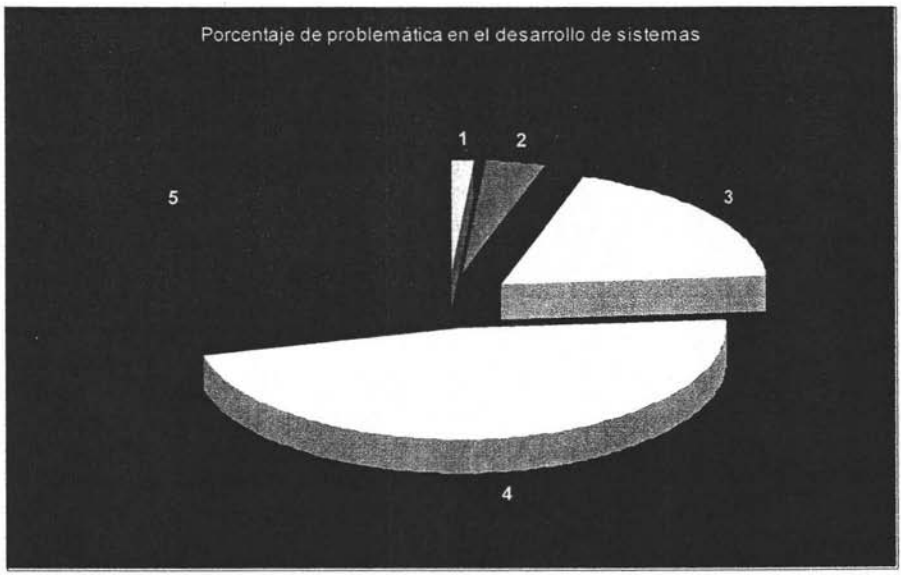


Ilustración 2"Representación Gráfica"

Al observar estos resultados observamos la enorme importancia que se debe de considerar durante el análisis del mismo ya que en gran parte depende el éxito del sistema en donde el principal objetivo que un analista de sistemas debe de plantearse es el desarrollo de un sistema útil para las tareas que se necesita, sencillo de manejo y de implementación además de flexible para posible incorporación de nuevas necesidades.

2. Conceptos Básicos Web.

Para comprender el alcance de los sistemas Web se definirán aspectos básicos para comprender los mismos entre los cuales tenemos que un sistema Web ocupa la arquitectura Cliente Servidor en este área tenemos que definir lo que son los llamados "Hosts" y los llamados "clientes" en donde cada uno de los equipos de cómputo que participa en Internet recibe el nombre de computadora host. Algunos host sirven el contenido o las aplicaciones a otras computadoras, por lo que se les denomina servidores. Otras computadoras, consumen o utilizan el contenido y la información ofrecida por los servidores. A estas computadoras se les conoce como computadoras cliente. En conjunto, esta relación se denomina cliente/servidor.

Un programa cliente es aquel con una interfaz amigable con el usuario, el cual se ejecuta en su computadora y accede a los recursos de Internet. Cuando el programa cliente necesita algo de Internet, busca un programa servidor.

Un programa servidor envía de regreso una respuesta adecuada al programa cliente: por ejemplo, el navegador Web Netscape es un programa cliente y cada servidor Web de Internet es un programa servidor.

La manera de cómo se transmite la información desde el medio físico del cliente al servidor se realiza mediante el protocolo TCP/IP que es aquel que define las reglas de cómo se comunican y comparten información todas las máquinas conectadas a Internet

La arquitectura de TCP/IP tiene cuatro niveles jerárquicos:

Nivel de acceso a la red
Nivel de comunicación entre redes (IP)
Nivel anfitrión-anfitrión (TCP)
Nivel de Proceso o aplicaciones

En donde el Protocolo TCP (Transfer Control Protocol o protocolo de transferencia de control) se encarga de cumplir con el trasporte de los datos a través de la red, segmentando la información. Vigila que únicamente reciba los datos quien señaló el emisor, efectuando una codificación o encriptamiento de los mismos para evitar que algún otro elemento de la red, distinto al destinatario, pueda usarlos posteriormente, el protocolo IP. Define la nomenclatura que manejara cada computadora y consiste en un conjunto de 32 bits.

Por ejemplo la tabla de los 32 bits para la dirección 132.248.73.2 es la siguiente:

128	64	32	16	8	4	2	1	
1	0	0	0	0	1	0	0	132
1	1	1	1	1	0	0	0	248
0	1	0	0	1	0	0	1	73
0	0	0	0	0	0	1	0	2

Tabla 2 "Tabla de los 32 bits para la dirección 132.248.78.2"

Posteriormente tenemos el DNS (Domain Name Server) lo cual resulta particularmente muy útil para identificar direcciones especificas en el Web ya que es difícil hablar usando números y aun más recordarlos, pero es la única forma en que las máquinas se pueden diferenciar en Internet
Por lo cual las direcciones IP se le asignan nombres para que puedan ser recordados.
Quién se encarga de hacer esta conversión es una máquina con un software específico y es llamada DNS (Servidor de Nomenclatura de Dominios)

La estructura de un sistema de dominios es separada por puntos

Sintaxis: computadora.subdominio.organización.dominio

Computadora jorge
Subdominio cdf
Organización difesa
Dominio mx
Ejemplos: jorge.difesa.com.mx
mail.banamex.com
www.altavista.com

Los dominios por organización se subdividen en:
•edu Educativos
•gob Gubernamentales
•com Comerciales
•org Organizaciones
•net Proveedores de Acceso a Internet

En todos los países excepto Estados Unidos a parte se le agrega dos letras al dominio estas son las del país tomándose generalmente la primera letra y la siguiente consonante
México mx
Argentina ar

Quien se encarga de definir estos nombres es una autoridad llamada NIC la cual existe en cada país y se encarga de administrar los dominios que le corresponden a su país por ejemplo el NIC México administra los dominios com.mx, edu.mx, org.mx, net.mx Por lo cual podemos ver que existen dos tipos de Direcciones en Internet:
•Por Dominio (jorge.difesa.com.mx)
•Por dirección IP (192.168.1.100)

La forma en que se hace esta conversión es la siguiente

192	Mx
168	difesa
1	Cdf
100	Jorge

Tabla 3 "Correspondencia de acuerdo a la dirección IP".

También es importante conocer los llamados Servicios en Internet ya que Internet provee diferente servicios los cuales cuentan cada uno con protocolos de comunicación diferentes entre si los principales son:

telnet
ftp
correo electrónico
usenet o News
IRC
http

Estos servicios utilizan puertos de comunicación predeterminados algunos de ellos son:

25 Correo Electrónico
23 Telnet
22 Secure Shell
21 FTP
80 HTTP
1900 Usenet o News

Nuestra base de estudio para el sistema es el WWW (World Wide Web) el cual es un medio de organizar y presentar la información para trabajar en Internet, consta de una enorme variedad de documentos almacenados en computadoras alrededor del mundo. Está basado en la tecnología del hipertexto y fue desarrollado en los laboratorios de la CERN, en Europa, con la finalidad de tratar de facilitar en acceso al usuario al compartir información sobre proyectos de investigación. Más tarde, para ponerlo a la disposición del público en general, se le enriqueció con un formato visual e intuitivo.

Cabe señalar que el WWW trabaja sobre una estructura cliente-servidor y en el Web tenemos que diferenciar los conceptos básicos como sitio Web el cual es una colección de páginas de la Web mantenidas por una universidad, escuela, gobierno, compañías o particulares y una pagina Web es un documento en la Web, el cual puede incluir texto, imagen, sonido y video (multimedia) la cual se puede accesar por medio del Servidor Web , que es una computadora conectada a la Internet que hace precisamente a las páginas de la Web accesibles al mundo.

3. Tecnología De Tres Capas.

3.1. Descripción.

La tecnología de tres capas principalmente tiene como objetivo aislar su funcionamiento interno de tal manera que únicamente intercambien datos para su procesamiento u almacenamiento según sea el caso. De tal manera que proporcione una independencia física con respecto a la lógica. Se ha seleccionado esta tecnología para la implementación del sistema en estudio debido a que proporciona ventajas muy importantes como los son:

- Ahorro de tiempo para mantenimiento, cambios, actualizaciones, modificaciones y cualidades de diseño.
- Protección de la información física en caso de falla de la aplicación lógica.
- Facilidad para migrar la información a tecnologías nuevas o emergentes.
- En caso de modificaciones mayores no es necesario regenerar al 100 % toda la aplicación en conjunto dado que permite aprovechar elementos de la misma.

La siguiente figura ilustra de qué manera se divide la metodología de tres capas:

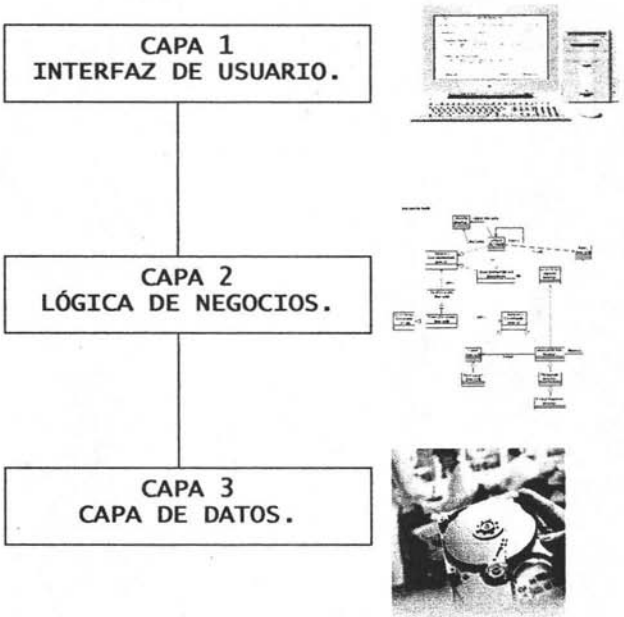


Ilustración 3 "Esquema general de la tecnología de tres capas."

En la cual la capa de interfaz de usuario son las diferentes pantallas y componentes por lo cual el interactuara con el sistema, la capa de lógica de negocios son todos los procesos y/o reglas empleadas para llevar a cabo las operaciones del negocio así como las políticas del mismo, finalmente tenemos la capa de datos que es donde se almacenara físicamente la información, encargándose esta de llevar el control sobre la misma y su recuperación.

4. Descripción De Software Libre Empleado.

4.1. Definición.

Inicialmente tenemos que marcar la diferencia de "Software Libre" y "Software Gratis", al primero dado un programa o conjunto de programas se le denomina "Software Libre" cuando "Se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software. De modo más preciso, se refiere a cuatro libertades de los usuarios del software:

- La libertad de usar el programa, con cualquier propósito (libertad 0).
- La libertad de estudiar cómo funciona el programa, y adaptarlo a las necesidades (libertad 1). El acceso al código fuente es una condición previa para esto.
- La libertad de distribuir copias (libertad 2).
- La libertad de mejorar el programa y hacer públicas las mejoras a los demás. (libertad 3). El acceso al código fuente es un requisito previo para esto.

Así un programa es software libre si los usuarios tienen todas estas libertades. Así pues, se tiene la libertad de distribuir copias, sea con o sin modificaciones, sea gratis o cobrando una cantidad por la distribución"[2] de tal manera se diferencia lo que es "Software Libre "y "Software Gratis" (Freeware: Este termino se proporciona al software que es propietario y se permite su distribución libre pero no su modificación del mismo).

La forma tradicional de distribución y comercialización de software de aquellos productos de amplio uso tales como sistemas operativos, procesadores de texto, bases de datos, compiladores, etc. Se basa principalmente cuando una empresa productora de software distribuye un producto de este tipo, solamente entrega al comprador una copia del programa ejecutable, junto con la autorización de ejecutar dicho programa en un número determinado de computadoras.

En el contrato que suscriben ambas partes, comúnmente denominado *"licencia"* del producto, queda expresado claramente que lo que el cliente adquiere es simplemente la facultad de utilizar dicho programa en determinada cantidad de computadoras (dependiendo del monto que haya pagado). En este sentido, la licencia deja en claro que el programa sigue siendo propiedad de la empresa productora del mismo y que el usuario no está autorizado a realizar ningún

[2] Definición proporcionada por Freesoftware Foundation Inc. 2001

cambio en él ni tampoco a analizarlo para determinar como realiza sus funciones. Entre otras cosas, el usuario también tiene el problema de que no podrá solucionar la mayoría de los errores que pudiese descubrir en el programa. Por otra parte, la empresa productora deslinda toda responsabilidad respecto a las pérdidas que pudiera ocasionar para el comprador el uso del software en cuestión. Esto incluye los daños por fallas del producto, debidas a errores del mismo. En este sentido, la licencia aclara que el vendedor no garantiza la aplicabilidad del producto a ningún fin determinado ni mucho menos se hace responsable de probables perdidas de información a causa del producto (Software con Licencia) .

4.2. Software Empleado.

4.2.1. Justificación.

Se ha seleccionado software libre para la implementación del sistema debido a que se podrá utilizar sin ninguna restricción para en la organización, de tal manera no generara un costo excesivo para el pago de licencias, por otra parte existe una cantidad considerable de contras para escoger "Software Propietario" como lo son:

Dependencia de un proveedor: Como se menciono anteriormente, la corrección de errores o el agregado de nuevas funciones en un programa solamente puede hacerse si se dispone de la licencia u autorización por parte de la empresa productora de software. Es claro que, al ser la empresa proveedora la única que dispone de dicha autorización, sólo esta puede atender a los requerimientos de un cliente insatisfecho con el producto del cual ha adquirido una licencia de uso.

Esto pone al usuario en una clara situación de dependencia del proveedor. Si el programa presenta algún defecto, éste debe aceptar las condiciones de la empresa productora del programa (en el supuesto caso de que dicha empresa reconozca el error y acceda a repararlo) o en el caso que se desee realizar cambios mayores simplemente la empresa lo realizaría con un costo considerable o finalmente no se realizaría por ser cambio específico.

Así se ha seleccionado por su funcionalidad, estabilidad y alto rendimiento los siguientes programas:

Sistema Operativo (SO).- Se ha seleccionado Linux como Sistema Operativo ya que Linux es una versión de UNIX de libre distribución (UNIX es uno de los sistemas operativos más populares del mundo debido a su extenso soporte y distribución. Originalmente fue desarrollado como sistema multitarea con tiempo compartido en mainframes a mediados de los 70's y desde entonces se ha convertido en uno de los sistemas mas utilizados gracias a su seguridad y su estabilidad más aun la mayoría de las versiones de UNIX son muy caras.) inicialmente desarrollada por Linus Torvalds en la Universidad de Helsinki, en Finlandia este fue desarrollado con la ayuda de muchos programadores y expertos de UNIX a lo largo y ancho del mundo, gracias a la presencia de Internet.

Cualquier persona puede acceder a Linux y desarrollar nuevos módulos o cambiarlo a sus necesidades. El núcleo de Linux no utiliza ni una sola línea del código de propietario de AT&T o de cualquier otra fuente de propiedad comercial, y buena parte del software para Linux se desarrolla bajo las reglas del proyecto de GNU de la Free Software Foundation, Cambridge, Massachusetts. De todas las posibles versiones disponibles de Linux se ha seleccionado Linux Red Hat 9. Respecto a este sistema operativo podemos mencionar las siguientes ventajas: es muy estable (el sistema raramente presentara errores que impida su funcionamiento), es confiable (debido a que continuamente se van corrigiendo errores), es seguro (el sistema tiene un esquema de seguridad muy bueno), no requiere hardware costoso, además de que cuenta con buena cantidad de soporte.

Lenguaje de Interacción Cliente-Servidor.- En este caso se ha seleccionado PHP, donde PHP es el acrónimo de Profesional Home Pages el cual fue creado por Rasmus Lerdorf a finales de 1994 aunque propiamente no fué prácticamente utilizable hasta principios de 1995 esta primera versión fue conocida como "personal Home Page Tools", al principio solo estaba compuesto por algunas macros que facilitaban el trabajo a la hora de crear una pagina Web. Hacia mediados de 1995 se creo el analizador sintáctico y se llamo PHP/F1 Versión 2 y solo reconocía el texto HTML y algunas directivas de mSQL. A partir de aquí la contribución al código fue completamente pública. Desde entonces el crecimiento de PHP ha sido exponencial por sus funcionalidades y alta seguridad así como la interacción con bastantes DBMS como lo son Oracle, PosgreSQL, MySql, Sybase, Informix, etc. Por ello ha sido seleccionado para implementarlo en nuestro proyecto por su estabilidad, seguridad, compatibilidad, robustez, flexibilidad y la capacidad de ser accedido por una cantidad considerable de equipos.

Servidor Web.- Como servidor Web se ha seleccionado "Apache Web Server" , Apache es el servidor Web más popular desde Abril de 1996 ya que las estadísticas en Diciembre del 2003 se encontró que el 57% de los sitios Web de Internet estaban usando Apache , haciendo de este el más utilizado en comparación con el resto de servidores Web juntos[3] .

El proyecto Apache http Server es un esfuerzo para desarrollar y mantener un servidor http Open Source (Software Libre) para diversos Sistemas Operativos, tales como Linux. La finalidad de este proyecto es la de proporcionar un servidor seguro, eficiente y extensible que proporcione servicios http también seguros y eficientes

Es por ello que se selecciono Apache por todas sus cualidades que presenta y a sus prácticamente nulos errores.

Manejador de Bases de Datos (DBMS).- El manejador de Base de datos es la aplicación que aísla la capa física de las otras capas, con esto tenemos mayor seguridad en nuestra información. Para este apartado se ha seleccionado MySQL el cual cuenta con 2 tipos de licencias una comercial y una libre, esta última será la indicada para nosotros para las necesidades de la

[3] Estadistica hecha por Netcraft Web Server Survey.

organización. Este manejador de base de datos es un manejador Relacional de tal manera cuida la integridad de los datos así como nos provee cualidades bastante buenas como lo es la estabilidad, la seguridad y otras ad hoc a las otras tecnologías utilizadas.

Es importante mencionar que el DBMS MySQL tiene sus fundamentos en la teoría del álgebra relacional, teoría creada por el Dr. Edgar F. Codd esta teoría se fundamenta en teoremas matemáticos bien formados y comprobables, de tal manera la estructura relacional de nuestro DBMS esta comprobada matemáticamente por lo que se trato de un producto de Jure y no Facto esto es en base a un teoría comprobada se ha creado el software.

Así pues al proveer a la organización con software libre de alta calidad no solo estamos creando un software de calidad si no que representara un ahorro de gasto para la misma debido a la naturaleza del mismo que no es necesario contratar licencias muy costosas.

5. Teoría De Inteligencia Artificial Para La Optimización De Rutas.

5.1. Introducción A La Inteligencia Artificial.

La computación empezó a surgir como una ciencia y se dieron cuenta que los robots podían realizar tareas mucho más complejas de lo que ellos imaginaban. Se interesaron en el concepto del "razonamiento humano" y se dieron cuenta que si pudieran "aprender" de su medio, se podría llevar a cabo el hecho de crear vida artificial, y de esta manera hacer que los robots pensaran y pudieran razonar. Se comenzó por desarrollar algoritmos capaces de resolver problemas específicos, se interesó en aplicar la lógica matemática en la resolución de dichos problemas, y es aquí donde comenzó la IA mas sin embargo actualmente seremos capaces de responder la pregunta ¿ Podemos crear robots, computadoras que sean capaces de razonar como un humano?, pues a la fecha nadie ha sido capaz de responderla con bases científicas pero tampoco nadie ha sido capaz de negarla con bases científicas y en base a los diferentes estudios que se han realizado durante el transcurso de la historia todo parece indicar a que si es posible. La mejor forma para entender lo que es Inteligencia Artificial es definiendo por principio lo que es inteligencia.

Inteligencia (según el diccionario de la lengua española): Es la capacidad para aprender o comprender. Suele ser sinónimo de intelecto (entendimiento), pero se diferencia de éste por hacer hincapié en las habilidades y aptitudes para manejar situaciones concretas y por beneficiarse de la experiencia sensorial.

Inteligencia (según la psicología): Capacidad de adquirir conocimiento o entendimiento y utilizarlo en situaciones novedosas. En condiciones experimentales se puede medir en términos cuantitativos el éxito de las personas a adecuar su conocimiento a una situación o al superar una situación específica.

Podemos definir la IA. como *"el resultado de las maneras en las cuales las computadoras o robots pueden mejorar las tareas cognoscitivas, en las cuales, actualmente, la gente es mejor"*. De esta manera, encontrar la mejor manera de resolver problemas de matemáticas, encontrar la ruta optima para llegar a un objetivo específico, son parte del razonamiento humano, y que hasta ahora el hombre ha deseado poder imitarla en la INTELIGENCIA ARTIFICIAL. Otra definición es la de indicar la capacidad de un artefacto (llámese computadora, robot o de otra índole) de realizar los mismos tipos de funciones que caracterizan al pensamiento humano. La posibilidad de desarrollar un artefacto así ha despertado la curiosidad del ser humano desde la antigüedad. Con el avance de la ciencia moderna la búsqueda de la IA ha tomado dos caminos fundamentales: la investigación psicológica y fisiológica de la naturaleza del pensamiento humano, y el desarrollo tecnológico de sistemas informáticos cada vez más complejos que SIMULAN la inteligencia del hombre.

5.2. Clases De Complejidad En Las Ciencias Computacionales.

La complejidad se ha dividido principalmente en dos clases; como son las clases P y NP. La clase P corresponde a los problemas cuyos algoritmos de solución son de complejidad polinomial deterministas en tiempo; y los problemas NP son los problemas cuya solución hasta la fecha no han podido ser resueltos de manera exacta por medio de algoritmos deterministas eficientes, pero que pueden ser resueltos por algoritmos no-deterministas y cuya solución son de complejidad polinomial en tiempo.[4]

En donde P lo definimos como "La colección de todos los problemas de decisión los cuales tienen algoritmos deterministicos en tiempo polinomial "y NP denota "La colección de todos los problemas de decisión los cuales tienen algoritmos de solución no deterministicos en tiempo polinomial". [5] Los algoritmos No Deterministicos los podemos interpretar como algoritmos que siempre tienen un camino exitoso, o dicho de otra manera se puede obtener una solución "optima" que requiere tiempo polinomial, sobre esto podemos denotar que P es subconjunto de NP. Nuestro Problema a solucionar cae directamente en el tipo NP ya que solo podemos obtener la solución mas optima pero no necesariamente la mejor de todas.

5.3. Definición Del Problema.

Nuestro problema de optimización de rutas se le conoce como El Problema del Agente Viajero el cual es uno de los Problemas de Optimización Combinatoria más tradicionales y estudiados este consiste en una distribución de n puntos a visitar (puntos de distribución) en un plano bidimensional. Cada punto es alcanzable desde cualquier otro punto mediante un camino cuya longitud viene definida por la distancia euclídea que separa a ambos puntos.

---

[4] Juan Luis Castro Peña . Departamento de Inteligencia Artificial Universidad de Granada

[5] D. R. Stinson ; An Introduction to the Design and Analysis of Algorithms ; Second Edition ;Canada 1987

Una unidad de distribución se dedica a recorrer estos puntos entregando productos. El interés es el de recorrer todas las puntos empleando para ello el trayecto total más corto posible.
De modo más concreto, el problema consiste en hallar una ruta que, partiendo de un punto determinado, recorra todos los puntos restantes, volviendo al punto de partida y minimizando la distancia total del recorrido.
Las restricciones que se han de tener en cuenta en este problema son las siguientes:

- Cada punto sólo puede ser visitado una única vez en el recorrido.
- El recorrido debe pasar por todos los puntos.
- El recorrido debe ser único (no puede haber varios recorridos inconexos).

Este problema si lo observamos como un grafo lo podemos definir como: "Un grafo a cuyas aristas se le ha asignado un pesos entre los vértices que representan las ciudades que debe visitar el agente viajero que pueden representar kilometraje, costo, tiempo o algún otra cantidad que se desee minimizar "[6]. El objetivo principal es encontrar la ruta mínima que pase por cada punto exactamente una vez y regrese a la ciudad inicial. En resumen la meta es encontrar un circuito hamiltoniano que minimice la suma de los pesos de las aristas. [7]

Así podemos Definir el siguiente ejemplo:

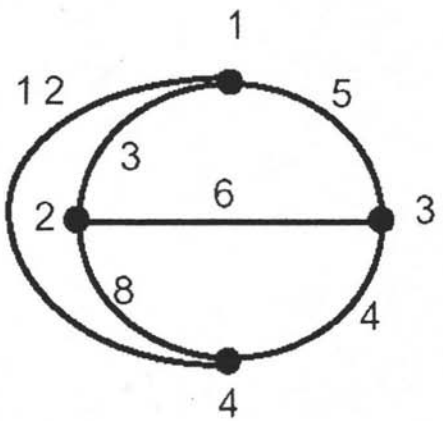


Ilustración 4 "Ejemplo de un grafo con pesos definidos."

En donde se desea recorrer el grafo de manera que toque todos los vértices sin que se repitan tomando el punto inicial al vértice 1 y en este mismo vértice es

[6] Kenneth A. Ross y Charles R.B. Wright ; Matemáticas Discretas , Prentice Hall 1990
[7] Castañeda Roldan C.Y. ;Estudio Comparativo de Solución del PAV; UDLA, 2000

necesario terminar el recorrido , se obtienen rutas y con ello un costo , finalmente se detecta el costo mínimo cuyo análisis es el siguiente:

Ruta	Nodos	Costo
1	1,2,3,4,1	3+6+4+12=25
2	1,2,4,3,1	3+8+4+5=20
3	1,4,2,3,1	12+8+6+5=31
4	1,4,3,2,1	12+4+6+3=25
5	1,3,2,4,1	5+6+8+12=31
6	1,3,4,2,1	5+4+8+3=20

Tabla 4 " Análisis de costos de recorridos."

Así del presente ejemplo se observa que el recorrido de costo mínimo es 30 aunque hay 2 rutas que hacen lo mismo es indistinto dado que los puntos son los mismo recorridos al revés[8].

5.3.1. Complejidad.

Este problema es considerado representativo de problemas cuya resolución pertenece al dominio de extrema complejidad. Esto se debe a que no se conoce ningún algoritmo capaz de generar la solución óptima sin recurrir a la enumeración sistemática de soluciones. Para darse una idea de la complejidad observemos la siguiente tabla:

O(1)	Constante	No depende del tamaño del problema	Eficiente
O(log n)	Logarítmica	Búsqueda binaria	
O(n)	Lineal	Búsqueda lineal	
O(n•log n)	Casi lineal	Quick-sort	
O(n2)	Cuadrática	Algoritmo de la burbuja	Tratable
O(n3)	Cúbica	Producto de matrices	
O(nk) k>3	Polinómica		
O(kn) k>1	Exponencial	Algunos algoritmos de grafos	Intratable
O(n!)	Factorial		

Tabla 5 "Ejemplos de complejidad computacional de algoritmos."

Para un problema de tamaño n (n puntos) la cantidad de soluciones que se han de explorar viene descrita por un factorial de n. Lo que esto implica se puede apreciar fácilmente en la siguiente tabla que confronta diversos tamaños de problemas con el tamaño del espacio de búsqueda y con el tiempo que requeriría una computadora promedio capaz de procesar 1 millón de soluciones por

[8] Castañeda Roldan C.Y. ;Estudio Comparativo de Solución del PAV; UDLA, 2000

segundo. Se produce una explosión combinatoria tal, que resulta impensable obtener una solución óptima a partir de n=30.

Puntos	Soluciones	Tiempo	Unidad
4	3	0,000003	segundos
5	12	0,000012	segundos
6	60	0,00006	segundos
7	360	0,00036	segundos
8	2520	0,00252	segundos
9	20160	0,02016	segundos
10	181440	0,18144	segundos
11	1814400	1,8144	segundos
12	19958400	19,9584	segundos
13	239500800	3,99168	minutos
14	3113510400	51,89184	minutos
15	4,3589E+10	12,108096	horas
16	6,5384E+11	7,56756	días
17	1,0461E+13	121,08096	días
18	1,7784E+14	5,6355272	años
19	3,2012E+15	101,43949	años
20	6,0823E+16	19,273503	siglos
21	1,2165E+18	385,47006	siglos
22	2,5545E+19	8094,8713	siglos
23	5,62E+20	178087,17	siglos
24	1,2926E+22	4096004,9	siglos
25	3,1022E+23	98304117	siglos
26	7,7556E+24	2,458E+09	siglos
27	2,0165E+26	6,39E+10	siglos
28	5,4444E+27	1,725E+12	siglos
29	1,5244E+29	4,831E+13	siglos
30	4,4209E+30	1,401E+15	siglos
31	1,3263E+32	4,203E+16	siglos
32	4,1114E+33	1,303E+18	siglos
33	1,3157E+35	4,169E+19	siglos

Tabla 6 "Estimado de tiempo en procesar posibles Rutas por Métodos Deterministas"

5.4. Teoría De Grafos.

5.4.1. Definición

Un Grafo lo podemos definir como un diagrama que si se interpreta de forma adecuada proporciona información. Se puede distinguir 2 tipos de grafos los dirigidos y los no dirigidos cuya diferencia radica en que el primero lleva una dirección y en el otro no. Los elementos de un grafo son sus objetos y sus líneas en donde los objetos se representan por puntos y reciben el nombre de vértices o nodos y las líneas que los unen se llama aristas, se denotan los vértices de un Grafo G como
$v_1, v_2, v_3, \ldots\ldots, V_v$. y las aristas por $e_1, e_2, e_3, \ldots\ldots, e_e$, siendo el subíndice v de v_v el número total de vértices y el subíndice e de e_e el número total de aristas . Cuando una arista conecta a un vértice consigo mismo se le denomina lazo.[9]

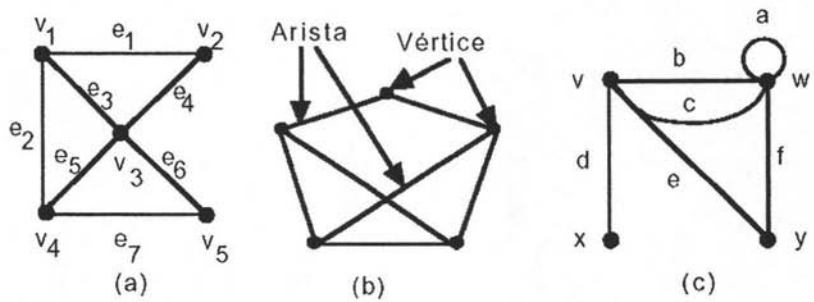


Ilustración 5 " Ejemplo de Vértices y Aristas"

De esta manera si observamos nuestro problema de la distribución de rutas mediante GRAFOS podemos integrar a nuestras necesidades el hecho de ver a cada vértice como un punto de distribución y a cada arista como el trayecto entre 1 punto de distribución a otro así formando rutas de distribución.

Más sin embargo la solución mediante grafos se complica considerablemente por la naturaleza de la organización por lo que solo nos sirve como marco teórico para la visualización grafica del problema.

[9] Kenneth A. Ross y Charles R.B. Wright ; Matematicas Discretas ;Prentice Hall , 1990

5.5. Problemas De Optimización.

Para los problemas de optimización contamos con diferentes formas de solucionarlos. Más sin embargo cada método tiene sus características propias. Acotaremos en 2 familias los algoritmos para su estudio lo cual son: Algoritmos Exactos y Algoritmos de Aproximación.

Es importante definir adecuadamente el concepto de algoritmo:

Un algoritmo es una serie finita de pasos para resolver un problema.

Hay que hacer enfásis en dos aspectos para que un algoritmo exista:

1) El número de pasos debe ser finito. De esta manera el algoritmo debe terminar en un tiempo finito con la solución del problema.
2) El algoritmo debe ser capaz de determinar la solución del problema.

Durante el desarrollo y estudio de los diferentes algoritmos se mencionara la palabra heurística esta palabra de acuerdo con ANSI/IEEE Std 100-1984, la heurística trata de métodos o algoritmos exploratorios durante la resolución de problemas en los cuales las soluciones se descubren por la evaluación del progreso logrado en la búsqueda de un resultado final. Se suele usar actualmente como adjetivo, caracterizando técnicas por las cuales se mejora en promedio el resultado de una tarea resolutiva de problemas. En otras palabras la heurística nos sirve para mejorar métodos en cuanto al tiempo de ejecución esto implica que usando la heurística en algoritmos no puede garantizar que nos proporcionara la solución exacta pero si la aproximación más cercana a ella y esto en un tiempo computable bastante razonable y eficiente.

Lo interesante de la solución de estos problemas es precisamente que si son algoritmos de aproximación, ¿qué tanto se aproximan a su solución real y cuál es un valor garantizado de exactitud?, para ello existe una métrica teórica estándar para medir la calidad de un algoritmo de aproximación, que es el cociente para el peor de los casos que indica que tan cerca del valor óptimo están las soluciones encontradas por un algoritmo de aproximación. Dicha cercanía puede ser expresada como el cociente del valor obtenido por el algoritmo de aproximación sobre el valor de la solución óptima. Se usará la notación OPT(I) para denotar el valor óptimo de la función objetivo en la instancia I y V(I,A(I)) para denotar el valor obtenido por el algoritmo A al evaluar la instancia I en la función objetivo.

$$C_A = \min\left[\frac{V(I,A(I))}{OPT(I)}, \frac{OPT(I)}{V(I,A(I))}\right]$$

5.6. Algoritmos Exactos.

5.6.1. Fundamentos.

Generalmente cuando programamos una computadora para la solución de problemas como por ejemplo contestar preguntas como : ¿Cuántos caminos hay para.....?, ¿Listar todas las posibles soluciones para...?, ¿Hay un camino para...?, preguntas que surgen muy comúnmente en las diferentes materias de nuestra carrera , nos surge la inquietud de determinar si se pueden solucionar y se llega a la conclusión de que usualmente requiere de una búsqueda exhaustiva dentro del conjunto de todas las soluciones potenciales, por eso los algoritmos que resuelven este tipo de problemas reciben el nombre de algoritmos exactos o de búsqueda exhaustiva. Por ejemplo, si se desean encontrar todos los números primos menores de 10^4, no hay método conocido que no requiera de alguna manera, el examinar cada uno de los números enteros entre 1 y 10^4.
De otra manera si se desean encontrar todos los caminos de un laberinto, se deben examinar todos los caminos iniciando desde la entrada[10].

Se estudiaran dos técnicas para organizar tales búsquedas. La primera "retroceso" (backtracking), ésta trabaja tratando continuamente de extender una solución parcial. En cada etapa de la búsqueda, si una extensión de la solución parcial actual no es posible, se "va hacia atrás" para una solución parcial corta y se trata nuevamente. El método "retroceso" es usado en un amplio rango de problemas de búsqueda, incluyendo el analizar gramaticalmente (parsing), juegos, y calendarización (scheduling). La segunda técnica, es la búsqueda exhaustiva ingenua que es el complemento lógico de "retroceso" en que se tratan de eliminar las no-soluciones en lugar de tratar de encontrar la solución. Este método es útil principalmente en cálculos numéricos teóricos[11].

Se debe tener en mente, sin embargo, que estos métodos son sólo técnicas generales. Su aplicación resultará en algoritmos cuyos requerimientos en tiempo son prohibitivos; en general la velocidad de las computadoras no es práctica para una búsqueda exhaustiva de más de 108 elementos. Así, para que estas técnicas sean útiles, ellas deben considerar solamente una estructura dentro de la cual se aproxima el problema. La estructura debe ser hecha a la medida, a generalmente con una buena función heurística, para cuadrar con el problema particular así que el algoritmo resultante será de uso práctico.

5.6.2. BackTracking (Algoritmos de Retroceso).

La idea fundamental de "retroceso" puede ser más fácilmente entendida en el contexto de un juego de laberinto: Se inicia en algún cuadro específico con la meta en algún otro cuadro especificado por una secuencia de movimientos desde un cuadro al próximo cuadro. La dificultad es que está restringido por la existencia de barreras que prohíben ciertos movimientos, como se ilustra en la figura siguiente.

---

[10] Edward M. Reingold, Jurg Nievergelt, Narsingh Deo ; Combinational Algorithms Theory and Practice ; Prentice Hall , Inc. , Englewood Cliffs , New Jersey 07632. 1977
[11] Ibidem

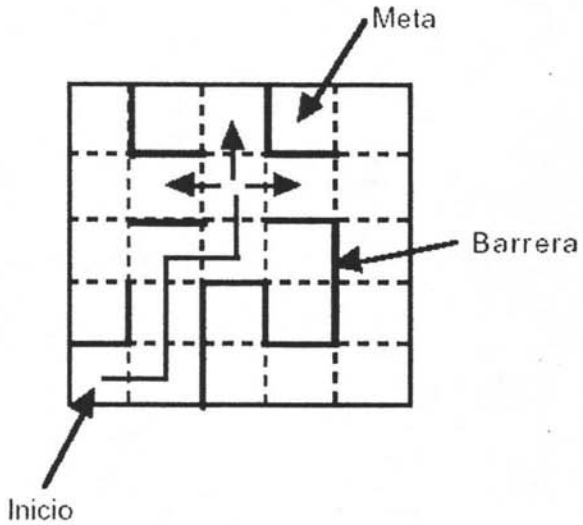


Ilustración 6 "Recorrido de un laberinto"

Así un camino para recorrer el laberinto es viajar desde un cuadro de inicio hasta un cuadro final llamado meta de acuerdo a dos reglas:
1. Desde el cuadro actual, tomar cualquier camino no explorado previamente.
2. Si el cuadro actual no tiene caminos inexplorados, volver hacia atrás un cuadro en el último camino que se siguió para encontrar caminos viables de explorar.
La primera regla dice como extender el camino actual, si es posible, y la segunda regla dice como regresar cuando se ha estancado. Esta es la esencia del método de "retroceso": expandiendo la solución actual tanto como sea posible; y cuando la solución ya no pueda ser expandida, regresar y tratar una alternativa en el estado más reciente para el cual hay una alternativa no probada.

De esta manera se podrá generar un árbol de acuerdo a las posibles soluciones y podremos elegir de todas ellas la más adecuada, sin embargo este método no resulta viable para nuestro problema ya que no representa un ahorro significativo de tiempo de búsqueda por la cantidad de posibles puntos de distribución a visitar.

5.6.3. Búsqueda Exhaustiva Ingenua.

El algoritmo de Búsqueda Exhaustiva Ingenua consta de un conjunto N de puntos llamados nodos, (puntos de distribución) estos son:

$$N = \{1, 2, ..., n\}$$

Donde n es el número total de nodos en el grafo generado (número de vértices o número de puntos de distribución), y cuyo objetivo será generar todas las posibles soluciones y tomar las más cortas. S es un conjunto ordenado de nodos el cual incluye un camino parcial (que contiene una lista ordenada de k enteros) y la suma de los pesos de sus arcos (los pesos son unidades cuantificables como el tiempo ponderado con respecto al kilometraje por recorrer) :

S = (k, [i1, i2, ..., ik], peso)

Donde W(i, j) es la matriz de adyacencia o de costos que representa un problema de puntos de distribución (o de venta) , (esto es una matriz dónde se puede apreciar cual es el costo de un vértice o punto de distribución a otro) un ejemplo de una matriz de adyacencia para 4 puntos de distribución se muestra a continuación:

De / al	*V1*	*V2*	*V3*	*V4*
V1	0	3	5	12
V2	3	0	6	4
V3	5	7	0	2
V4	8	1	4	0

Tabla 7 "Ejemplo de una matriz de costos para 4 puntos"

Así se genera un circuito y se compara el peso de ese circuito con el peso del circuito inicial que recibe el nombre de Best_S_soFar. (Esta notación es la más generalizada para identificar nodos) . Si el circuito recién calculado resulta ser mejor entonces este es nuestro nuevo conjunto Best_S_soFar, de lo contrario se retiene el anterior. El proceso se detiene cuando ya ningún circuito mejora en peso al del conjunto Best_S_so_Far.

Los procedimientos que realizan la tarea de búsqueda exhaustiva son dos el primero de ellos que se llama Búsqueda_Exhaustiva, este es el encargado entre otras cosas de calcular el peso inicial y de llamar a Búsqueda_Exhaustiva_Rec. En Búsqueda_Exhaustiva_Rec recursivamente va generando el árbol de búsquedas hasta encontrar el costo mínimo.

Este algoritmo busca todos los (n - 1)! posibles caminos iniciando en 1, guardando el mejor. Hay mucho de paralelismo, porque después de k recursivos llamados a Búsqueda_Exhaustiva_Rec, hay (n - 1)*(n - 2)*.....*(n - k) subárboles independientes para buscar, el cual puede ser realizado en muchos procesadores.

Ya que los subárboles son igualmente largos se dice que el balance de carga es perfecto. El hecho de que se realice una búsqueda de todos los (n-1)! distintos caminos en el peor de los casos implica que tiene un O(n!).

Algoritmo que llama a Búsqueda_Exhaustiva_Rec

{01} Procedure **Búsqueda_Exhaustiva_Ingenua**
{02} peso = w(1, 2) + w(2, 3) + w(3, 4) + ... + w(n-1, n) + w(n, 1);
{03} Best_S_so_far = (n, [1, 2, 3, ..., n-1, n], peso);
{04} S = (1, [1], 0);
{05} Búsqueda_Exhaustiva_Rec (S, Best_S_so_far);
{06} Imprime_Resultados (Best_S_so_far);
{07} end de Búsqueda_Exhaustiva_Ingenua;

Algoritmo recursivo de Búsqueda Exhaustiva Ingenua

{08} Procedure **Búsqueda_Exhaustiva_Rec** (S, Best_S_so_far)

```
{09} Let S = ( k, [ i1, i2, ....., ik ], peso );
{10} Let Best_S_so_far = ( n, [ i1B, i2B, ....., inB ], pesoB );
{11} If k = n
{12} then
{13} new_peso = peso + A(ik, i1);
{14} if new_peso < pesoB
{15} then
{16} Best_S_so_far = ( k, [ i1, i2, ....., ik ], new_peso );
{17} end if;
{18} else
{19} for all j not in [ i1, i2, ....., ik ];
{20} new_peso = peso + A(ik, j);
{21} New_S = ( k + 1, [ i1, i2, ....., ik, j ], new_peso );
{22} Búsqueda_Exhaustiva_Rec ( New_S, Best_S_so_far );
{23} end for;
{24} end if;
{25} return
{26} end de Búsqueda_Exhaustiva_Rec;
```

En todos los algoritmos se numerarán las líneas en orden ascendente para poder Explicar en caso necesario alguna de ellas haciendo referencia al número. Así:

- {02} Calcular el valor inicial del peso *peso* , donde el peso depende del recorrido inicial que puede ser uno escogido arbitrariamente, así que en este método se escogió {1,2,3,4, ...n}, el cálculo del peso matemáticamente se representa como :

$$\text{peso}=A(n,1) + \sum_{i=1}^{m=n-1} A(i,i+1)$$

  se agrega el valor A(n,1) para que pueda cerrarse el circuito y su peso respectivo sea agregado.
- {03} Inicializar el circuito como:

  nodo(i) = i con i = 1,2,3,....., n

  donde n es el número total de nodos, vértices o de ciudades. El nodo(i) se deja en el registro Best_S_so_Far.
- {04} El registro S contendrá el circuito que se va formando en la recursividad, debe ser inicializado en (1, [1], 0) que significa que es el primer nodo, el conjunto que contiene la ruta que se va formando contiene al nodo No. 1 cuyo peso es 0.
- {05} Llamar al procedimiento de Búsqueda_Exhaustiva_Rec para la búsqueda del costo mínimo y la ruta respectiva.
- {09} y {10} Recibir la información que llega a las variables del procedimiento
- {11} Si k = n significa que ya se terminó de generar un circuito parcial, es decir deja de ser circuito parcial para ser circuito total y debe,
- {12} terminar de calcular el peso New_peso, al que se le sumará el peso que se encuentra en la posición A(S,nodo(k),1). Donde k es el último nodo del

circuito.La columna debe ser la 1 para cerrar el circuito al nodo 1, porque este fue el inicial.

- {14} a - {17} Guardar en el registro New_S ← S siempre y cuando el peso New_peso recién calculado sea menor que el peso pesoB anexando los datos recién calculados.
- {19} a - {24} Se crean los circuitos con sus respectivos pesos y repetir la operación en forma recursiva.

5.6.4. Ramificación y Acotamiento Ingenuo y Mejorado.

El método Ramificación y Acotamiento ingenuo es también llamado Naive Branch and Bound. Este se obtiene al hacerle una mejora al método de Búsqueda Exhaustiva Ingenua, esta consiste en hacer una "poda" o corte al árbol de búsqueda. Así se observa cada una de las rutas parciales y si alguna de ellas ya es mayor que la de la mejor solución encontrada con anterioridad no hay razón para seguir buscando por ese camino. Por lo tanto se evita seguir en él y esto es lo que se llama poda del árbol de búsqueda.

Este método se perfila como uno de los mejores y no tan complicados para su implementación como el posible candidato a implementar en nuestro proyecto. A continuación veremos los Algoritmos de Aproximación mas sin embargo debido a ala complejidad de su análisis solo se hará un pequeño estudio de los mismos ya que estos no representan una utilidad extras para la naturaleza del problema que se busca solucionar [12].

5.7. Algoritmos de Aproximación.

Muchos problemas NP son formulados en forma natural como problemas de optimización. Se sabe que la programación dinámica y los algoritmos de Búsqueda Ingenua que son algoritmos exactos para encontrar soluciones óptimas requieren tiempo superpolinomial a pesar de que estos algoritmos son más eficientes que la búsqueda exhaustiva pura, su tiempo de corrida es aún exponencial. Por esa razón es entonces natural buscar otra alternativa de solución como son los algoritmos de aproximación que trabajan en tiempo polinomial.

Es importante mencionar que la clase de problemas de optimización que se derivan de problemas de decisión en NP es llamada NPO, como un acrónimo para designar que son problemas de optimización derivados de problemas NP[13].

Lo interesante de la solución de estos problemas es precisamente que si son algoritmos de aproximación, ¿qué tanto se aproximan a su solución real y cuál es un valor garantizado de exactitud?, para ello existe una métrica teórica estándar para medir la calidad de un algoritmo de aproximación, que es el

---

[12] Articulo CS267. Assignament 5 : Traveling Salesman Problem , Due March 21, 1995.

[13] Jonson D.S . Aproximation Algorithms for Combinational Problems. Journal of Computer and Systems Sciences . Vol. 9 , pp 256-278, 1974

cociente para el peor de los casos que indica que tan cerca del valor óptimo están las soluciones encontradas por un algoritmo de aproximación. Dicha cercanía puede ser expresada como el cociente del valor obtenido por el algoritmo de aproximación sobre el valor de la solución óptima. Se usará la notación OPT(I) para denotar el valor óptimo de la función objetivo en la instancia I y V(I,A(I)) para denotar el valor obtenido por el algoritmo A al evaluar la instancia I en la función objetivo o heurística

Para un mejor entendimiento de la exactitud tomaremos esta definición: "Un algoritmo A es un algoritmo de aproximación para un problema de optimización Π en NPO si dada una instancia de entrada I, éste calcula una solución factible S para cada entrada I [14]

Así podemos obtener el grado de aproximación de los algoritmos denotado por Π. Que inclusive varios autores toman como punto de clasificación de los algoritmos este parámetro (el grado de aproximación de la solución).

Iniciaremos nuestro estudio de los algoritmos de aproximación con los algoritmos genéticos que particularmente resulta extremadamente interesantes por el comportamiento y la analogía que estos poseen.

5.7.1 Algoritmos Genéticos.

La naturaleza nos enseña grandes maravillas, así los Algoritmos genéticos AGs (Genetic Algorithms) son una analogía de la estructura genética, es decir, basándose en los principios de selección natural se realizan procedimientos que simulan el comportamiento de los cromosomas dentro de una población, éste método ha resultado ser muy eficiente para buscar aproximaciones a mínimos globales en espacios grandes y complejos en relativamente poco tiempo.
Los componentes básicos de un algoritmo genético son:[15]

1. Operadores genéticos (mutación y cruzamiento)
2. Una representación apropiada del problema a resolver
3. Una función de Oportunidad o bien llamada de Aptitud (fitness)
4. Un procedimiento de inicialización.

El método AGs como preámbulo utiliza el procedimiento de inicialización para generar la primera población. Los miembros de la población a semejanza con los cromosomas son usualmente cuerdas de símbolos que permiten representar las posibles soluciones al problema que está tratando de ser resuelto. Simulando a los individuos de la población, estos compiten unos contra otros para ser usados en el proceso de reproducción. Es decir, cada uno de los miembros de la población es evaluado, y según su "Aptitud (fitness)" se le asigna una probabilidad de ser seleccionado para la reproducción. Matemáticamente hablando se usa esta probabilidad para que los operadores genéticos seleccionen algunos individuos, obteniendo de esta forma a nuevos individuos, donde generalmente son aquellos individuos con más éxito en cada competencia y que serán capaces de producir mayor descendencia que los que no tienen éxito.

---

[14] ibidem.
[15] Morrow M. Genetic Algorithms a new class of searching algorithms. Dr. Dobbs Journal . Abril 1991

El operador de cruzamiento de entre todos los individuos selecciona dos miembros de la población y combina sus cromosomas para generar a los mejores hijos. Se puede decir apoyando lo anterior que los genes de individuos buenos se propagan con más probabilidad a lo largo de la población, por esa razón dos padres buenos producirán alguna descendencia que se espera será mejor que la de sus antecesores. El operador de mutación selecciona un miembro de la población y modifica alguna parte de su cromosoma. Los elementos con peor "Aptitud (fitness)" son reemplazados por los nuevos individuos. Se espera que cada generación sucesiva se adapte mejor a su medio ambiente[16]

Los algoritmos Genéticos son diferentes a los métodos tradicionales como se describe en los siguientes cuatro puntos :

a) Trabajan codificando un conjunto de parámetros.
b) Buscan dentro de una población de puntos, no en un simple punto.
c) Usan una función objetivo que da información de rentabilidad y no se deriva de otros conocimientos.
d) Usan reglas de transición probabilísticas, no reglas deterministicas.

La función de aptitud no es más que la función objetivo de nuestro problema de optimización. El algoritmo genético únicamente maximiza, pero la minimización puede realizarse fácilmente utilizando el recíproco de la función maximizante (debe cuidarse, por supuesto, que el recíproco de la función no genere una división entre cero). Una característica que debe tener esta función es que tiene que ser capaz de "castigar" a las malas soluciones, y de "premiar" a las buenas, de forma que sean estas últimas las que se propaguen con mayor rapidez resultando su implementación peculiarmente complicado.

Los algoritmos Genéticos requieren de un conjunto de parámetros naturales de optimización que serán codificados como una cadena de longitud finita sobre algún alfabeto finito. La codificación más común de las soluciones es a través de cadenas binarias, aunque se han utilizado también números reales y letras.

El poder de los Algoritmos Genéticos proviene del hecho de que se trata de una técnica robusta, y pueden tratar con éxito una gran variedad de problemas provenientes de diferentes áreas, incluyendo aquellos en los que otros métodos encuentran dificultades. Si bien no se garantiza que el Algoritmo Genético encuentre la solución óptima del problema, existe evidencia empírica de que se encuentran soluciones de un nivel aceptable, en un tiempo competitivo con el resto de algoritmos de optimización combinatoria. En el caso de que existan técnicas especializadas para resolver un determinado problema, lo más probable es que superen al Algoritmo Genético, tanto en rapidez como en eficacia. El gran campo de aplicación de los Algoritmos Genéticos se relaciona con aquellos problemas para los cuales no existen técnicas especializadas. Incluso en el caso en que dichas técnicas existan, y funcionen bien, pueden efectuarse mejoras de las mismas hibridándolas con los Algoritmos Genéticos.[17]

---

[16] Ibidem
[17] Morrow M. Genetic Algorithms a new class of searching algorithms. Dr. Dobbs Journal . Abril 1991

El Algoritmo Genético Simple, también denominado Canónico, se representa a continuación en Pseudo código :

{01} **Algoritmo_Genético_Simple;**
{02} Begin
{03} Generar una población Inicial;
{04} Calcular la función de evaluación de cada individuo;
{05} While NOT Terminado do
{06} Begin // Producir una nueva generación
{07} For Tamaño_población /2 do
{08} Begin // Ciclo Reproductivo
{09} Seleccionar dos individuos de la generación anterior, para el cruce (probabilidad de selección proporcional a la función de evaluación del individuo);
{10} Cruzar con cierta probabilidad los dos individuos obteniendo dos descendientes;
{11} Mutar los dos descendientes con cierta probabilidad;
{12} Calcular la función de evaluación de los dos descendientes mutados;
{13} Insertar los dos descendientes mutados en la nueva generación;
{14} End del for;
{15} If la población ha convergido
{16} then
{17} Terminado = True;
{18} end del if
{19} end del while
{20} end del Algoritmo_Genético_Simple;

Así se observa que se necesita una codificación ó representación del problema, que resulte adecuada al mismo. Además se requiere una función de ajuste ó adaptación al problema, la cual asigna un número real a cada posible solución codificada. Durante la ejecución del algoritmo, los padres deben ser seleccionados para la reproducción, a continuación dichos padres seleccionados se cruzarán generando dos hijos, sobre cada uno de los cuales actuará un operador de mutación. El resultado de la combinación de las funciones anteriores será un conjunto de individuos (posibles soluciones al problema), los cuales en la evolución del Algoritmo Genético formarán parte de la siguiente población[18].

Esta forma de tratar o de coneptualizar nuestro problema resulta de una complejidad muy alta y no representa ventaja sobre otros considerando dicha complejidad , mas sin embargo es realmente interesante su análisis y estudios para comprender la forma de cómo se puede tratar un problema a tal grado de simular la forma biológica de la evolución.

---

[18] Algoritmos Geneticos P. Larrañaga Departamento de Ciencias de la Computación e Inteligencia Artificial Universidad del Pais Vasco – Euskal Unibertsitatea 1999.

5.7.2. Método Dos Optimal.

Esta técnica de solución también recibe el nombre de "2-Opt" que es la forma corta de "Dos Optimal", o bien "2-Operadores" u " Opción Doble". Esta es una de las heurísticas más exitosas para obtener una solución cercana al problema de la distribución de la forma PAV.

Este inicia con un ciclo Hamiltoniano llamado H, esta ruta inicial para el PAV puede ser arbitraria o bien (1, 2, ..., n). Se borran r aristas de H, produciendo así r caminos desconectados (algunos de los cuales pueden ser nodos aislados). Se reconectan estos r caminos de tal forma que producen otra ruta para el PAV llamada H', en la que se usan aristas diferentes de aquellas que fueron removidas de H. Así H y H' difieren una de la otra en exactamente r aristas; las (n-r) aristas remanentes son las aristas en común[19].

La siguiente figura ilustra el método 2-Opt.

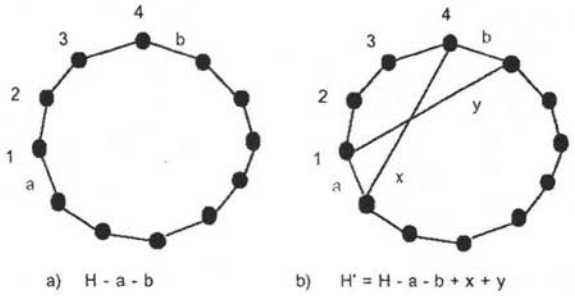


Ilustración 7 "Ejemplo de unión de vértices para formar un camino 2-Opt"

Así se calcula el peso total w(H') de la ruta H', y si w(H') < w(H), reemplazar H con H' y repetir el proceso; de otra manera, tomar otro conjunto de r aristas de H para intercambiarlas. Tales intercambios (de conjuntos de r aristas) se continúan hasta que ninguna otra mejora se pueda realizar por intercambio de r aristas. La solución final es llamada r-Optimal o r-Opt, en este caso se estudiará con r = 2 siendo así el Dos-Optimal o 2-Opt[20]. El procedimiento de intercambio de aristas terminará en un óptimo local (no necesariamente en un óptimo global), produciendo de esta manera una solución aproximada. Este algoritmo de aproximación para el problema de optimización en la distribución de rutas ilustra una aproximación para resolver muchos problemas de optimización, que se conocen como búsqueda local o búsqueda por cercanía. Este método de mejoras sucesivas en la ruta del PAV puede ser usada para en general, entre mas alto sea el valor de r en el procedimiento de intercambio, mejor será la solución. Pero el gasto computacional también se incrementa rápidamente con el valor de r.

---

[19] Jose Arreola . Notas de la Materia Analisis y Diseño de Algoritmos UDLA. 1996

[20] Jose Arreola . Notas de la Materia Analisis y Diseño de Algoritmos UDLA. 1996

Existirán: $\begin{pmatrix} n \\ r \end{pmatrix}$

Subconjuntos de r aristas en un ciclo de con n aristas .Intercambiar cada uno de estos pares de aristas una vez requiere tiempo O(nr). Por lo que se debe analizar correctamente lo que se desea, es decir entre la exactitud de la solución y el gasto computacional del tiempo de proceso para seleccionar la mejor ruta de distribución. Si r es dos se trata del método 2-Opt y tendrá por lo tanto un O(n2).

Finalmente existen otros métodos que no se adecuan a nuestro problema por ser de otra naturaleza como son el método Adaptación Prim y las Redes Neuronales Artificiales, ya que están orientados a la solución de otra clase de problemas.

5.7.2. Método Adaptación Prim.

El método Adaptación Prim se adecuó al PAV, porque el Algoritmo de Prim, produce un árbol generador mínimo o árbol de expansión mínimo o árbol abarcador mínimo T (Minimum spanning tree) que es un grafo no dirigido conectado con aristas ponderadas. Inicialmente se escoge un vértice arbitrario y se permite que T sea un árbol que consiste de ese vértice seleccionado. Se repite la etapa de selección del siguiente vértice n-1 veces, tomando en cuenta cual arista tiene el costo mínimo con exactamente un punto final en T, incluyendo ese vértice en T. Se dice que es un algoritmo voraz porque siempre añade la arista de costo mínimo y suprime la mayor. Además hace elecciones mínimas manteniendo simultáneamente el subgrafo conexo y acíclico. Más aún, no necesita que las aristas del grafo *G* estén ordenadas de antemano. La adaptación consiste en tomar los vértices del árbol abarcador mínimo y producir un ciclo hamiltoniano.

Sea *Best_S_so_Far* el conjunto de vértices en el momento en el que el algoritmo se detiene. Sean V(T) los vértices del subgrafo *T* (árbol de expansión mínimo). Sean *w* el peso de la suma de las aristas ponderadas del subgrafo T y *a* la matriz de adyacencia del grafo G. El procedimiento forma un árbol abarcador mínimo. En cada etapa el algoritmo busca una arista de menor peso que una un vértice en T con un nuevo vértice fuera de T. Entonces añade esa arista y ese vértice a T y repite el proceso. [21] El algoritmo se describe a continuación:

{01} Function **Adaptación_Prim**
{02} begin
{03} Se inicializa el vector col a ceros;
{04} menor_en_fil ←•;

[21] Castañeda Roldan C.Y. ;Estudio Comparativo de Solución del PAV; UDLA, 2000

```
{05} dir ←0;
{06} i ← 0;
{07} Best_S_so_Far[0] ← 0;
{08} col[0] ← 1;
{09} for j ← 1 to n_vertices - 1 do
{10} begin
{11} menor_en_fil ←•;
{12} for k ←0 to n_vertices -1 do
{13} begin
{14} if (col[k] = 0) and (menor_en_fil > a[Best_S_so_Far[i]][k])
{15} begin
{16} menor_en_fil ← a[Best_S_so_Far[i]][k];
{17} dir ← k;
{18} end del if;
{19} end del for k;
{20} Best_S_so_Far[j] ←dir;
{21} col[Best_S_so_Far[j]] ← 1;
{22} i ← j;
{23} w ← w + menor_en_fil;
{24} end del for j
{25} w ← w + a[Best_S_so_Far[n_vertices-1]][0];
{26} return w;
{27} end de la function Adaptación_Prim
```

Se explicarán las líneas más importantes:

{03} Se lleva un registro para cada vértice *x* en *V(G)* del vértice *u* en *a* con menor valor *a(u,x)* en el vector *col* que es inicializado en ceros. Cuando se localice un vértice adecuado la posición correspondiente se marcará con 1.
{04} y {10} Para localizar el menor de cada fila de la matriz de adyacencia se inicializa el menor_en_fil en •.
{07} Para ir guardando la ruta se usa el vector Best_S_so_Far que es inicializado con el nodo 0 en la dirección 0;
{08} El vector col es marcado con 1 porque ya se tomo el nodo 0 como punto de partida.
{09} El ciclo for corre desde 1 hasta n_vertices - 1 porque ya se tomo el nodo 0 anteriormente.
{12} a {19} Se busca la arista con menor valor en peso.
{20} a {21} se localiza la dirección de esa arista y se guarda en el vector Best_S_so_Far marcando al vector col con 1 en la dirección de Best_S_so_Far[j].
{23} Se actualiza el peso sumándole a este el menor_en_fil localizado.
{25} Aquí se adapta para que se cierre el camino Hamiltoniano, sumando al peso el peso de la matriz en la dirección de Best_S_so_Far[n_vertices-1]][0].

El tiempo para encontrar la x más cercana en *a* y después actualizar registros es O(n). Pero finalmente en el peor de los casos corre en un tiempo $O(n^2)$ por los dos ciclos anidados.

Usando el algoritmo anterior, se realiza una descripción más detallada del método Adaptación Prim para el PAV de 5 ciudades que se muestra en la Figura siguiente:

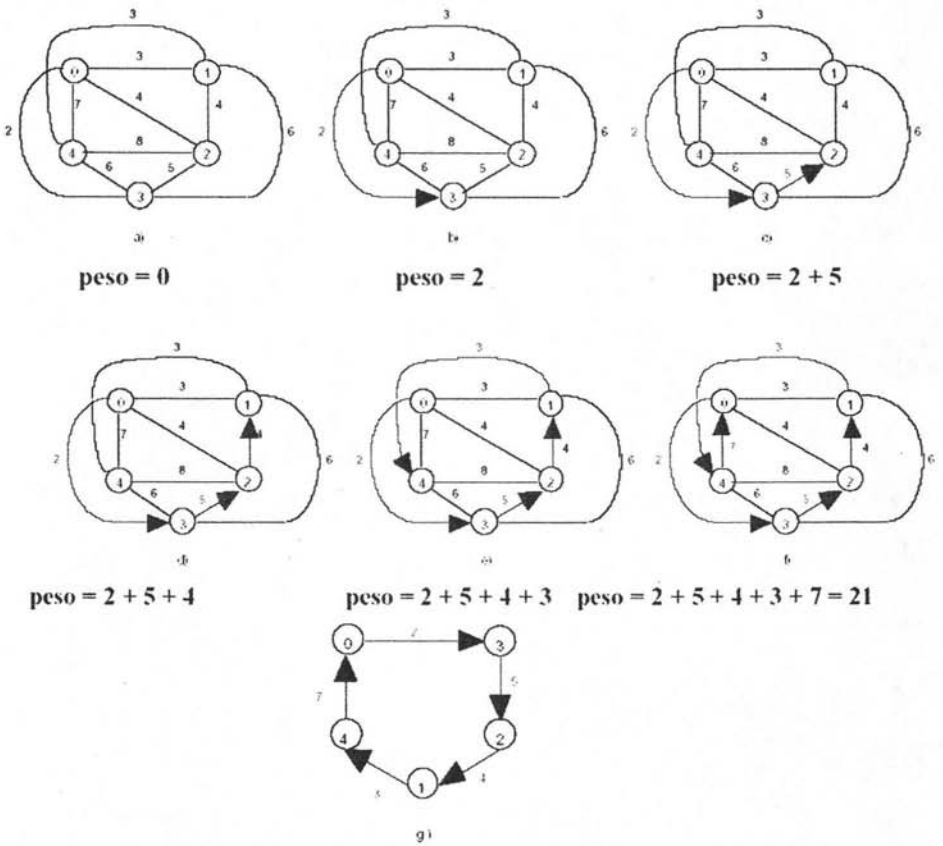


Ilustración 8 "Ruta óptima por medio de adaptacipón prim de un problema PAV"

Se parte de la Figura "a", puede iniciarse en cualquier nodo, pero en este caso será el nodo 0. Se busca la arista con el menor peso que salga del nodo 0, siendo el nodo 3 con un peso de 2. Y se repite la operación, recordando que un nodo que ya fue seleccionado no puede participar nuevamente. Como se muestra en la Figura "a" a "g".

En la Figura "d" hay dos números de peso mínimo con valor 4, pero el nodo 0 no puede seleccionarse porque primero formaría un ciclo lo cual no es permitido

en el árbol abarcador de Prim, y segundo se repetiría el nodo 0 cosa no permitida en el PAV.

Por lo que solo queda el mínimo 4 que va al vértice 1. En la Figura "f" se encuentra el peso mínimo y la ruta óptima, esta se reacomoda para que se distinga mejor en la figura "g". Además puede notarse que el resultado de la ruta óptima es 21 que es una aproximación al valor real que es de 19.

6. Teoría Para La Planeación De La Logística De Transporte.

Inicialmente para la planeacion en la logística de transporte tenemos que considerar los diferentes tipos de decisiones que se han de tomar en este caso tenemos a:

6.1. Tácticas Estratégicas De Optimización.

Diseño de Parque Vehicular o flota.- Se debe de tener un flota balanceada que cubre plenamente los requisitos para los fines que conlleva, esto implica que no solamente es necesario llevar a cabo la distribución de los productos en los tiempos establecidos sino que los productos arriben al punto de distribución en prefectas condiciones, lo cual se logra teniendo contenedores aislados y secos protegidos de las inclemencias del tiempo, también se debe de considerar no solo contar con los vehículos propios sino contar con una flota subcontratada a empresas externas , ya que esto nos llevará a asegurar el servicio a los clientes más importantes .

6.2. Operacionales.

En este punto es donde nos enfocaremos más ampliamente el análisis del mismo ya que nuestro fin es distribuir u optimizar los diferentes puntos de distribución con el menor costo financiero y de hora hombre. Esto sin duda conllevara a tener un crecimiento sostenido y a la vez se reducirá el impacto de los costos en los estados financieros de la empresa.

En general la operación por su naturaleza en tiempos y distancias de viaje se define como aleatoria dado que los diferentes contratiempos o eventos que se pueden presentar en la ciudad no son controlables, sin embargo , se puede proporcionar la aproximación más adecuada a la ruta optima , mediante dichos algoritmos. Cabe señalar que se tienen que considerar factores adicionales como: restricciones en la duración de la ruta (Por ejemplo cumplir con 8 horas de trabajo máximo marcado por la ley) , ventanas de tiempo (El punto de venta solo recibe en un intervalo definido de tiempo) , calles cerradas en cierto intervalos de tiempo a vehículos de carga, problemas de calles cerradas

6.3. Objetivos de Una Optimización de Operaciones.

Los objetivos que se lograran obteniendo operaciones sistematizadas y controles informáticos serán:

- Minimizar los costos reales.
- Minimizar los costos de flotas subcontratadas.
- Minimizar el tiempo de llegada.
- Maximizar la calidad del servicio (clientes atendidos a tiempo)
- Minimizar el riesgo de no cumplir la demanda solicitada por el cliente (en calidad y cantidad)
- Maximizar el beneficio NETO.

Esto nos llevara a tener un crecimiento y un soporte de clientes continuo y sostenido

7. Notación UML.

El *Unified Modeling Language* (UML) es una notación basada en diagramas que ha sido aceptada como estándar para describir sistemas de software orientados a objetos. Esta tecnología define varios tipos de diagramas que se utilizan para describir diferentes aspectos o vistas de un sistema. En UML, una de las herramientas principales para modelar comportamiento es la construcción Caso de Uso (*Use Case*) donde un Caso de Uso especifica una manera de usar un sistema sin revelar la estructura interna del mismo. Los Casos de Uso han sido adoptados casi universalmente para capturar los requerimientos de los sistemas de software; sin embargo, los Casos de Uso son más que una herramienta de especificación ya que tienen una gran influencia sobre todas las fases del proceso de desarrollo tales como el diseño, la implementación y las pruebas del sistema.

UML se puede usar para modelar distintos tipos de sistemas: sistemas de software, sistemas de hardware, y organizaciones del mundo real. UML ofrece nueve diagramas en los cuales modelar sistemas:

-Diagramas de Casos de Uso para modelar los procesos 'business'.
-Diagramas de Secuencia para modelar el paso de mensajes entre objetos.
-Diagramas de Colaboración para modelar interacciones entre objetos.
-Diagramas de Estado para modelar el comportamiento de los objetos en el Sistema.
-Diagramas de Actividad para modelar el comportamiento de los Casos de Uso, objetos u operaciones.
-Diagramas de Clases para modelar la estructura estática de las clases en el sistema.

-Diagramas de Objetos para modelar la estructura estática de los objetos en el sistema.
-Diagramas de Componentes para modelar componentes.
-Diagramas de Implementación para modelar la distribución del sistema.

Es importante señalar que UML es una notación no un método ya que no indica un proceso para modelar un sistema, sino cada modelador puede crear su metodología , inclusive algunas empresas desarrollan las mismas en base a la notación UML. Abarcando todos o algunos puntos de la notación.

El diagrama caso de uso nos resulta peculiarmente útil ya que nos da el punto de entrada para analizar los requisitos del sistema, y el problema que necesitamos solucionar en la organización o empresa.

Un caso de uso se modela para todos los procesos que el sistema debe llevar a cabo. Los procesos se describen dentro del caso de uso por una descripción con texto o una secuencia de pasos ejecutados. Los Diagramas de secuencia se pueden usar también para modelar escenarios gráficamente. Una vez que el comportamiento del sistema está captado de esta manera, los casos de uso se examinan y amplían para mostrar qué objetos se interrelacionan para que ocurra este comportamiento. Los Diagramas de Colaboración y de Secuencia se usan para mostrar las relaciones entre los objetos sin embargo no es muy común que los modeladores de sistemas los empleen pero dependen totalmente de los mismos.

Conforme se van encontrando los objetos, pueden ser agrupados por tipo y clasificados en un Diagrama de Clase. Es el diagrama de clase el que se convierte en el diagrama central del análisis del diseño orientado a objetos, y el que muestra la estructura estática del sistema. El diagrama de clase puede ser dividido en capas: aplicación, y datos, las cuales muestran las clases que intervienen con la interfaz de usuario, la lógica del software de la aplicación, y el almacenamiento de datos respectivamente. Los Diagramas de Componentes se usan para agrupar clases en componentes o módulos. La distribución general del hardware del sistema se modela usando el Diagrama de Implementación.[22]

Esta notación nos es particularmente útil para nuestra aplicación ya que como se menciono anteriormente, se implementara mediante la tecnología de tres capas así la notación UML nos permite traducir información desde múltiples modelos UML en código y estructura de bases de datos. Así se aprecia la utilidad y ventajas de fragmentar el sistema en capas.

Así el Diagrama de Clase se usa para generar una estructura base del código en el lenguaje escogido. Información de los diagramas de interacción, estado, y actividad, puede ofrecer detalles de la parte de los diferentes procedimientos del código de implementación.

La capa de datos del diagrama de clase se puede usar para implementar directamente el diagrama entidad-relación lógico para crear el diagrama físico representando las tablas y relaciones actuales de la base de datos relacional.

---

[22] Analisis y descripción de metodologia UML. IBM Racional Rose Sotware.

SEGUNDO CAPITULO "ANÁLISIS DE LA SITUACIÓN ACTUAL EMPRESARIAL."

1. Análisis Empresarial.

1.1. **Descripción De La Empresa**. La empresa en estudio se clasifica como pequeña por su cantidad de empleados, tan solo en México existen 2.9 millones de empresas las cuales el 99% corresponden a micro, pequeñas y medianas empresas (2.85 millones) [23] estas empresas constituyen un sector estratégico para el desarrollo económico y social del país ya que contribuyen con el 40 % de la inversión del PIB además de generar el 64 % de los empleos. El adecuado comportamiento macroeconómico de los últimos años no ha sido capaz de generar las condiciones necesarias para que PyMEs puedan crecer en forma dinámica y estable. Lo anterior , debido a que enfrentan una serie de dificultades que entorpecen su desarrollo integral como lo es:

1.1.1. Falta de internacionalización y desvinculación con los sectores mas dinámicos. El 94.3 por ciento de las exportaciones en México están concentradas en 312 grandes empresas nacionales y extranjeras[24] .

1.1.2. Falta de financiamiento. Las elevadas tasas de interés , acompañadas de las enormes restricciones de la banca comercial para los créditos trae como consecuencia que tan solo una cuarta parte de las PyMEs cuente con financiamiento de la banca comercial.

1.1.3. Falta de capacitación y barreras de acceso a tecnologías. Las empresas de menor tamaño carecen de mano de obra calificada y enfrentan importantes barreras de acceso a nuevas tecnologías por falta de información y recursos económicos.

1.1.4. Programas de apoyo a las PyMEs del gobierno federal. A pesar de que el gobierno Federal cuenta con una gran diversidad de apoyos para las PyMEs , la mayoría de los empresarios desconoce su existencia. El monto de recursos destinado para este fin , es muy pequeño en comparación con la magnitud de los retos que implican la modernización e incremento de la competitividad de este sector. La instrumentación de la política industrial es una labor compartida entre diversas dependencias y entidades del gobierno federal , lo cual dificulta que los programas del gobierno federal formaran un esquema de apoyo integral y coordinado.

---

[23] Fuente: Secretaria de Economía - Comisión Intersecretarial de Política Industrial.
[24] Fuente: Bancomext

De esta forma se puede apreciar a la tecnología que le permita a la empresa tener un crecimiento y a la vez ofrecer nuevos y mejores servicios mediante dicha tecnología, es muy difícil obtener los recursos económicos para adquirir productos tecnológicos propietarios. De esta manera se observa una ventaja muy importante al emplear software libre .

1.2. **Funciones y Operaciones de la Empresa.** Básicamente la operación de la empresa se puede limitar por 2 grandes áreas: el área operativa (proceso de distribución) y el área administrativa (análisis y gestión empresarial) de la primera su principal función la distribución de diferentes productos (libros, juguetes , cosméticos , revistas , periódicos , baterías, línea blanca , electrónicos , productos farmacéuticos, etc.) a diferentes puntos de venta ya sean cadenas comerciales de locales cerrados (WAL-MART, Comercial Mexicana , Carrefour , Sanborns , Gigante , Palacio de Hierro, etc.) o en domicilios particulares en el área metropolitana de la ciudad de México . Una vez realizada la entrega de los productos en el punto de distribución respectivo se obtiene la "evidencia" de entrega lo cual varia en función del mismo. Esto puede ser un número de folio como es el caso de las cadenas y consorcios (Sanborns, Comercial Mexicana, Gigante, etc.), sello y firma de recibido sobre documento (Como lo son entidades gubernamentales y locales establecidos) o simplemente la firma de la persona a la cual va dirigido el envió. Posteriormente a ello en función del resultado de la operación se realiza la notificación del área de almacén, lo que lleva a reportar los envíos como entregados, no entregados, rechazados o con devolución parcial. Otra de la función de la operación es la recuperación de la devolución del cliente para su acreditación al proveedor, la cual también se entrega al personal de almacén. Esta operación se lleva a acabo mediante la asignación de rutas fijas de los puntos de distribución, dichos puntos han sido asignados de acuerdo a la cultura del personal o como popularmente se menciona "mas o menos queda" esto trae una cantidad considerable de problemas que la empresa ha estado arrastrando por mucho tiempo lo que provoca perdidas económicas, recesiones de contratos con clientes y procesos administrativos deficientes. Respecto al área administrativa se encarga de las diversas actividades comunes de la empresa como lo son contabilidad, crédito y cobranza, atención a clientes, cuentas por pagar (Proveedores) y recursos humanos, además de las siguientes áreas que son particulares de acuerdo a la naturaleza de la empresa: Logística y distribución, control de trafico, circulación, control de almacén y atención a puntos de distribución. Estas áreas se encargan de llevar el control administrativo de acuerdo al resultado de la operación, sin embargo no existen políticas definidas ni control de formas, lo que facilita la propuesta de formatos y controles para un proceso ordenado y correcto.

1.3. **Situación Actual.** Actualmente se tiene un sistema administrativo deficiente en donde se basa la mayoría de las operaciones en registros no automatizados sin controles aplicados, por lo que se presenta una problemática muy importante de duplicidad de funciones e información poco confiable y no oportuna por lo cual existen una cantidad considerable de perdidas económicas y de horas hombre que esto conlleva dichas funciones se describen a continuación:

1.3.1. **Área de Comercializaron y Circulación.** Se tiene elaborada una base de datos en dbase IV con el listado de los clientes así como de las publicaciones , se emite diariamente la misma factura para los clientes mas sin embargo podrían presentarse cambios menores , así se imprimen dichas facturas sobre un formato ya definido por la empresa junto con una relación de las publicaciones contra punto de distribución para el control del área operativa una vez terminado este proceso se lleva a cabo la distribución de los elementos facturados a los puntos de distribución . Cabe señalar que las compras se realizan mediante dotaciones fijas de publicaciones , cuyo control se lleva mediante el uso de la herramienta EXCEL. Entre otras funciones esta área se encarga de informar y atender a los clientes y/o puntos de distribución sobre el estado de sus productos y devoluciones así como de informes adicionales salvo del estado del producto cuando se encuentre en el proceso de distribución en este caso pasa directamente al área de distribución.

1.3.2. **Área de Distribución.** Al arribar la mercancía al almacén se realiza la separación manual por ruta previamente asignada por el coordinador de trafico , así como de la selección de las publicaciones de acuerdo a la factura del punto de distribución posteriormente se realiza una relación con dichas entregas para llevarse a cabo la distribución, ésta relación es elaborada en la herramienta Excel , en caso que el cliente se comunique con la empresa para información sobre sus envíos se tiene que esperar al finalizar el día para verificar si fue entregado o no y al siguiente día se le informa sobre un posible percance o de su entrega correcta. Sin embargo uno de los problemas mas importantes de la empresa es que no se lleva un control adecuado sobre los envíos y esto provoca la perdida de mercancía o la ignorancia de la ubicación de la misma lo que provoca en el cliente, lógicamente, un enorme descontento ya que el quiere saber el resultado de sus envíos y la empresa es incapaz de determinar el mismo, hasta posteriormente haber realizado un investigación minuciosa.

1.3.3. **Cobranza.** La cobranza de las diferentes ventas llevadas a cabo se lleva de manera manual en registros escritos así como formas en la herramienta EXCEL, lo que lleva aumento de tiempo para el

funcionamiento óptimo del departamento y de la empresa por no tener un control de los cobros que se han realizado. Mas aun se tiene un problema muy grave el cual es la obtención de los saldos reales de los clientes , además de una discrepancia por varios miles de pesos con respecto a los registros del departamento contable lo que lleva a tener continuas revisiones exhaustivas de dichos registros para homogenizar los mismos, sin embargo después de cierto tiempo se vuelve a tener el mismo problema , ya sean por errores humanos en el proceso debido a la manipulación de la información directa por el personal .

1.3.4. **Control de Almacén.** El control de almacén resulta critico para la empresa dado que tiene la custodia de productos de un valor considerable de los clientes así como propios de allí, resulta fundamental tener un control minucioso sobre el mismo, y resulta totalmente lo contrario en la situación actual de la misma dado que no se tiene ningún registro de entradas ni de salidas de almacén solo las relaciones de embarqué que el cliente manda, lo que permite el fácil extravió de la mercancía. El funcionamiento del mismo radica en la recepción de la mercancía, la entrega de mercancía a las rutas y la recepción de devoluciones para su regreso a los proveedores, actividades que no tienen controles ni formas fijas solo notas realizadas manualmente sobre libretas.

1.3.5. **Proveedores.** La función de esta área se realiza mediante la consolidación de saldos de los proveedores (cuentas por pagar) mas sin embargo al no existir un sistema de control ,al igual que otras áreas usan sólo herramientas que tienen a su disposición como Excel se tiene una problemática similar la cual es que no se consolida eficientemente los saldos , a un grado tal que si la gerencia requiere conocer el saldo a pagar de un cliente tiene que esperar varios días para conocer el mismo , esto viola una de las reglas básicas y fundamentales de la información – La disponibilidad .

1.3.6. **Contabilidad y Recursos Humanos.** Las actividades de estas áreas se puede mencionar como las más sistematizadas de la empresa, dichos controles se llevan mediante herramientas comerciales propias para el área las cuales son: ASPEL-COI y ASPEL-NOI, sin embargo no se encuentra homogenizadas con las otras áreas lo que lleva una difícil sincronización de procesos entre estas áreas con las demás.

1.4. **Datos Básicos De Entrada.** A continuación se definirán los datos básicos de entrada sobre la operación:

Cliente.- Es el cliente del que viene el envío

Alfanumérico de Máximo 20 Caracteres.

Numero de Relación.- Número de Relación de Facturas

Numérico de Máximo 6 Caracteres.

Fecha de embarque.- Fecha de embarque a las instalaciones de Distribución.

Fecha En formato DD/MM/AAAA (Toma la Fecha Automáticamente Cuando se captura)

Numero de Factura .- Número no repetido de Factura

Numérico de 8 caracteres Máximo

Tienda.- Tienda a la que corresponde el envío .

Alfanumérico de 20 Caracteres Máximo.

Sucursal.- Sucursal a la que corresponde el envío

Alfanumérico de 20 Caracteres Máximo.

Valor Nominal de la Factura.- Costo del cliente a la Sucursal

Numérico con longitud de 6 caracteres.

Cajas y piezas que Consta la Factura.- Cantidad de cajas que lleva el envío.

Numérico de máximo 4 caracteres

Ruta de Distribución.- Ruta que lleva el Envío.

Numérico máximo de 2 caracteres

Fecha de Entrega.- Fecha en que se realiza la entrega de la Factura .

Fecha En formato DD/MM/AAAA

Folio de Entrega.- Número comprobatorio de entrega .

Alfanumérico de 20 caracteres

Nombre del Chofer que Entrega.

Alfanumérico de 20 caracteres.

Observaciones.- Alfanumérico de 40 caracteres.

El ingreso de la información al sistema será por medio de las relaciones de embarque que nos proporciona el cliente

Los tipos de formas que se utilizan son:

- Relaciones de embarque.
- Facturas.
- Cartas para Recoger Devolución.

Con estos datos básicos es posible la integración para el análisis del modelo de rutas así como su planeación estimada con el proceso de distribución.

TERCER CAPITULO "MODELO PROPUESTA PARA EL SISTEMA DE GESTIÓN Y CÁLCULO DE RUTA ÓPTIMA DE DISTRIBUCIÓN."

1. Definición Del Plan Para La Actualización Tecnológica.

El plan de actualización tecnológica comprende el desarrollo de soluciones a la medida para satisfacer el manejo de los siguientes aspectos dentro de la empresa.

- ✓ Promoción de servicios.
- ✓ Administración de clientes.
- ✓ Administración de servicios.
- ✓ Manejo de almacén y sus inventarios.
- ✓ Administración de vehículos y operadores.
- ✓ Logística de rutas (optimizadas) y control de envíos
- ✓ Facturación
- ✓ Control del personal operativo y administrativo.
- ✓ Información contable
- ✓ Información estratégica y de apoyo a la toma de decisiones.

Esta solución requiere de un conjunto de productos de Hardware y Software encaminados a resolver de manera confiable los problemas que se presentan en la organización para ello se implementara como sistema operativo Linux Red Hat versión 9 , y como servidor Web se empleara Apache, además de integrar la base de datos con MySql y la interacción de los equipos clientes con el equipo servidor será mediante PHP.

Nuestra propuesta de mejora no sólo contempla el área administrativa de la organización sino se busca hacer muchas más eficiente la operación mediante la correcta optimización de las rutas de distribución.

El modelo de integración del sistema de acuerdo al software libre seleccionado se presentara el diagrama de cómo interactuara la diferente infraestructura de telecomunicaciones de la organización así como las diferentes arquitecturas que son capaces de acceder al mismo, además de que permite la posibilidad de escalar o extender el alcance del mismo.

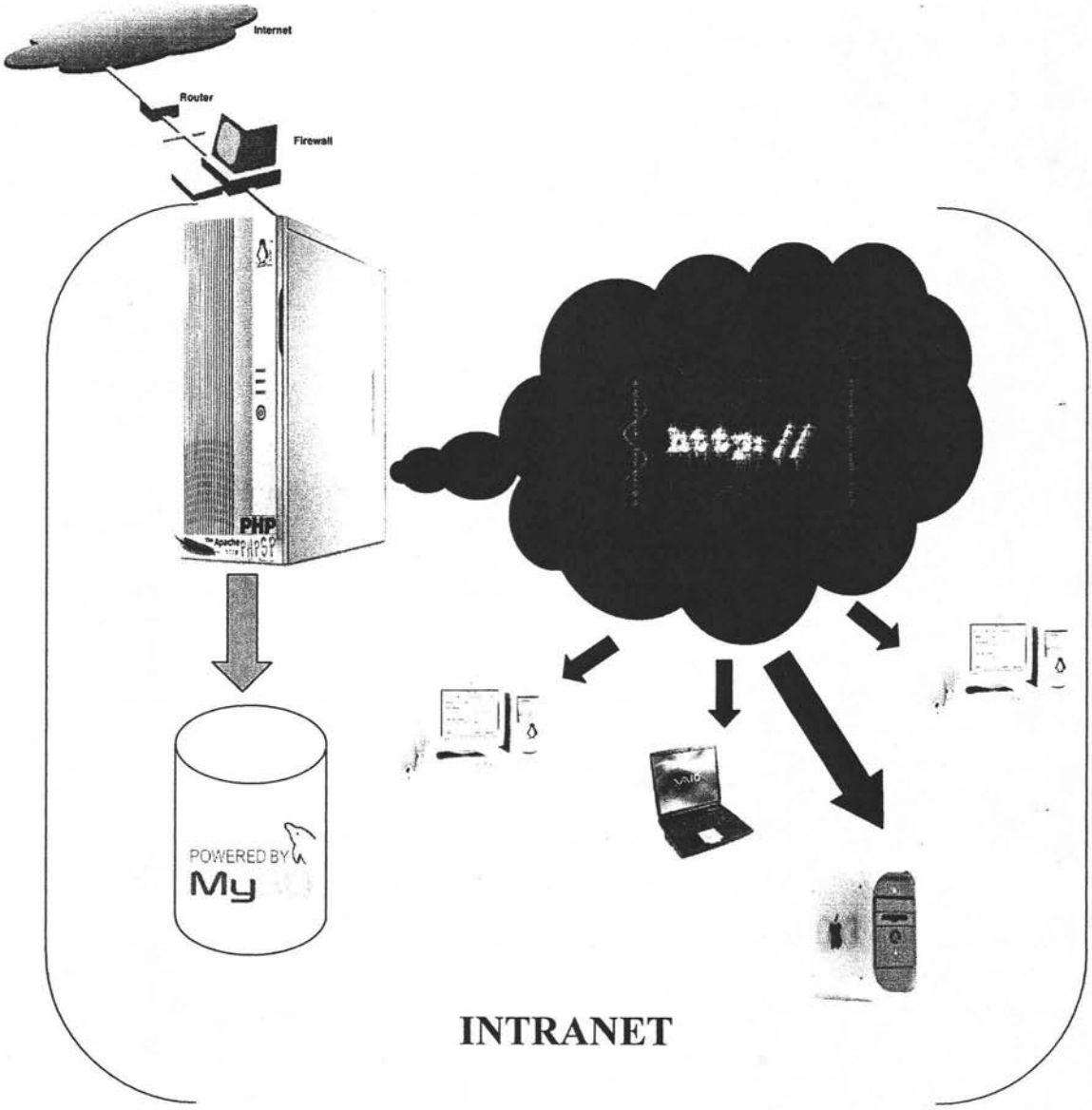


Ilustración 9 " Diagrama de integración del modelo"

2. Análisis Del Modelo Propuesto.

El modelo que se ha propuesto tiene el siguiente diagrama de casos de uso del cual se ha hecho análisis en función de los perfiles de puestos de la empresa, estos puestos generalmente tendrán muy pocos o nulos cambios sin embargo se podrá modificar directamente sobre el modelo entidad relación (base de datos) esta característica permite ser escalable a corto o largo plazo.

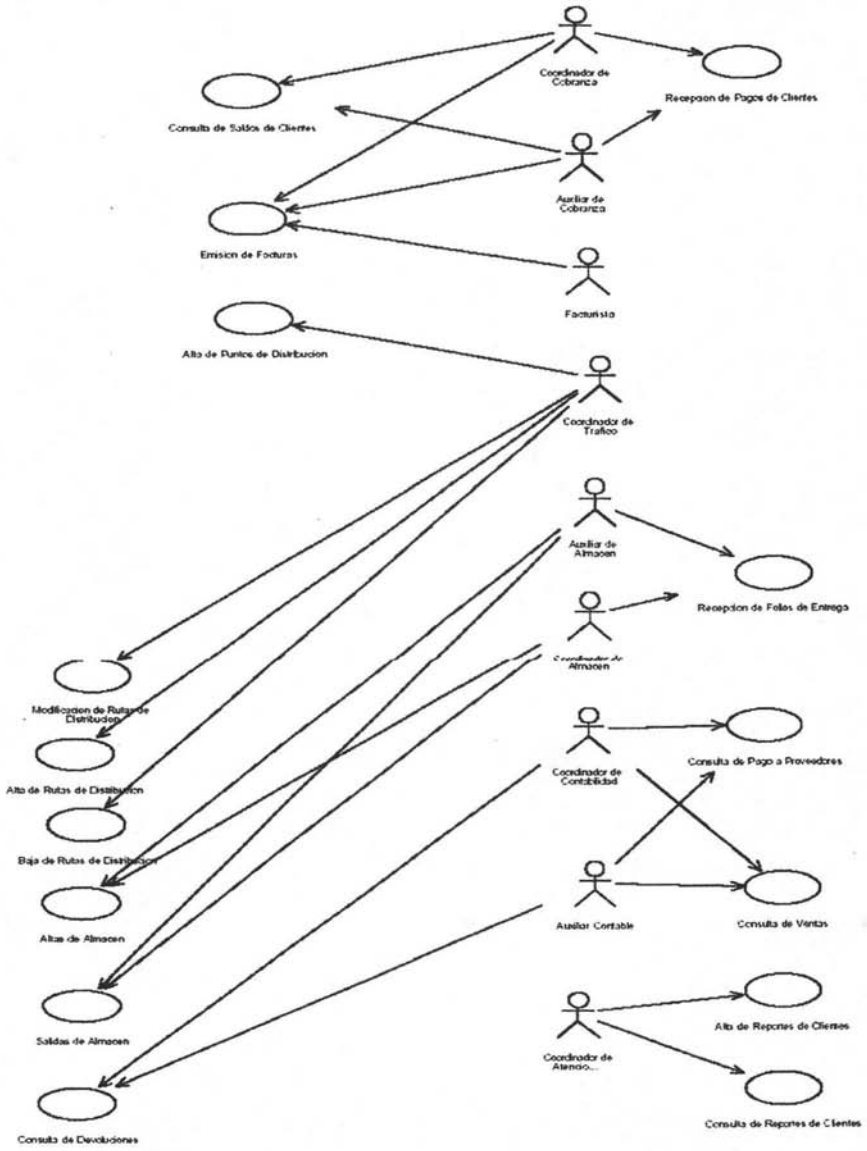


Ilustración 10 "Parte 1 del Diagrama de Casos de Uso del Sistema"

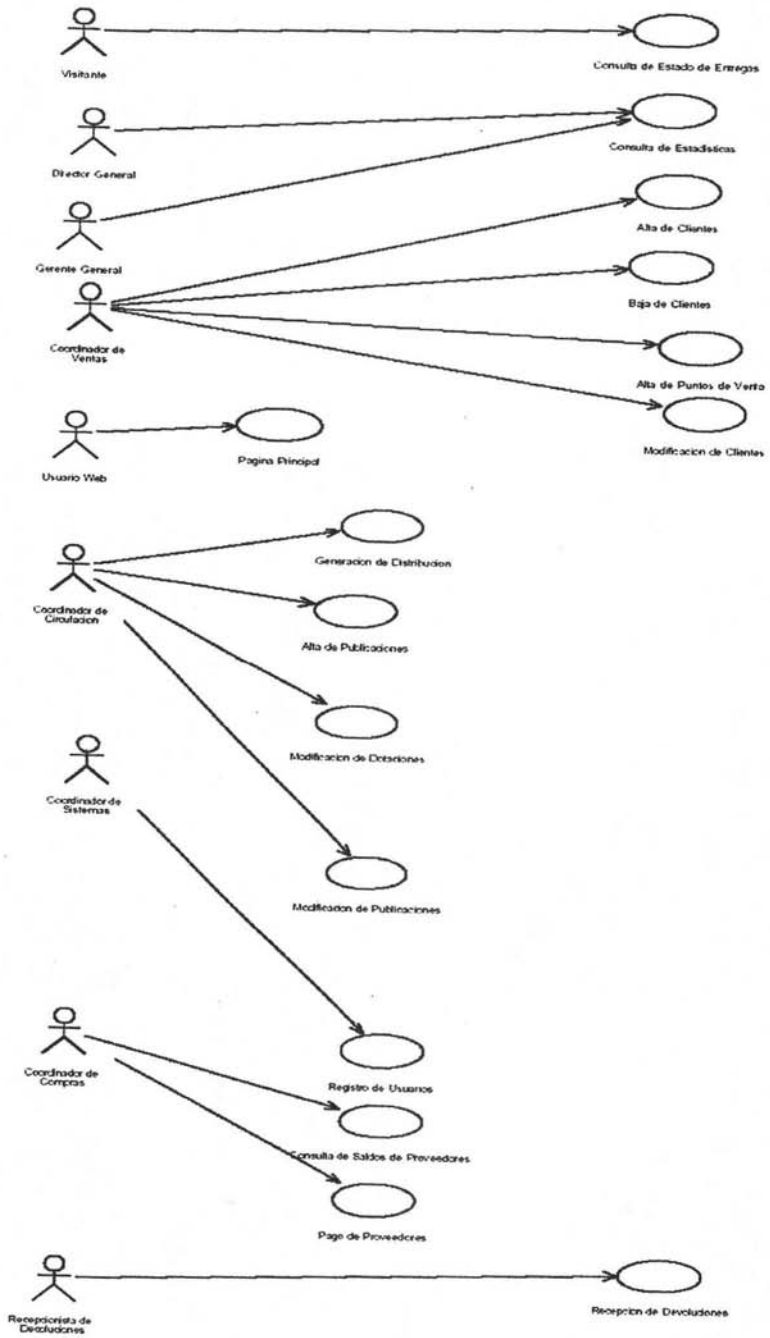


Ilustración 11 "Parte 2 del Diagrama de Casos de Uso del Sistema."

3. Desarrollo Del Procedimiento Para Obtener La Ruta Óptima.

Se han realizado pruebas para verificar el rendimiento de los algoritmos los resultados obtenidos son:

	TIEMPO EN MILISEGUNDOS RECORRIDO PARA LA CANTIDAD DE CIUDADES LISTADAS									
Metodo Aproximado. / Ciudades	3	4	5	6	7	8	9	10	11	12
Adaptación Prim.	2	0	1	1	1	0	0	0	1	1
2-Optimal.	1	0	1	1	1	1	2	3	4	6

	TIEMPO EN MILISEGUNDOS RECORRIDO PARA LA CANTIDAD DE CIUDADES LISTADAS									
Metodo Exacto./ Ciudades	3	4	5	6	7	8	9	10	11	12
Búsqueda Exhaustiva Ingenua.	2	2	7	19	27	2326	2155	219654	2475818	30972070
Ramificacion y Acotamiento Ingenuo Mejorado.	1	2	14	29	49	215	1478	4615	15106	78880

Tabla 8 " Analisis comparativo de tiempos."

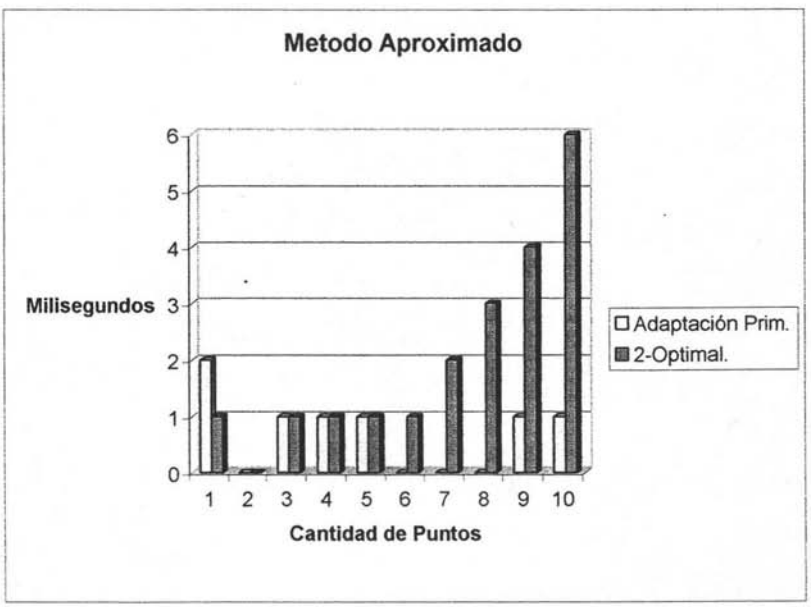


Ilustración 12 " Grafico de tiempos metodos aproximados."

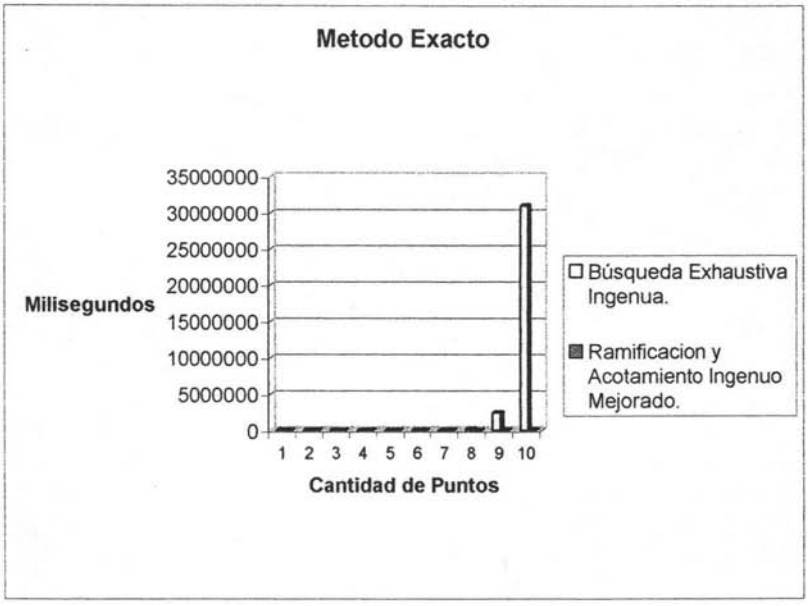


Ilustración 13 " Grafico de tiempos metodos exactos."

Se han realizado los procedimientos que realizan la tarea de búsqueda exhaustiva para una cantidad de hasta 9 puntos ya que es el método que más se adecua a nuestras necesidades con un tiempo de procesamiento adecuado ademas de que nos permite solucionar el problema con un resultado considerablemente bueno en un tiempo de computo apropiado y junto con el método de búsqueda modificado para tener flexibilidad para posibles cambios del método en cantidades. Adicionalmente para mas de 9 puntos se usará el adaptación prim. En el primer caso son dos procedimientos. El primero de ellos que se llama Búsqueda_Exhaustiva_Ingenua, es el encargado entre otras cosas de calcular el peso inicial y de llamar a Búsqueda_Exhaustiva_Rec.

{01} Procedure **Búsqueda_Exhaustiva_Ingenua**
{02} peso = w(1, 2) + w(2, 3) + w(3, 4) + ... + w(n-1, n) + w(n, 1);
{03} Best_S_so_far = (n, [1, 2, 3, ..., n-1, n], peso);
{04} S = (1, [1], 0);
{05} Búsqueda_Exhaustiva_Rec (S, Best_S_so_far);
{06} Imprime_Resultados (Best_S_so_far);
{07} end de Búsqueda_Exhaustiva_Ingenua;

{08} Procedure **Búsqueda_Exhaustiva_Rec** (S, Best_S_so_far)
{09} Let S = (k, [i1, i2, ..., ik], peso);
{10} Let Best_S_so_far = (n, [i1B, i2B, ..., inB], pesoB);
{11} If k = n

```
{12} then
{13} new_peso = peso + A(ik, i1);
{14} if new_peso < pesoB
{15} then
{16} Best_S_so_far = ( k, [ i1, i2, ....., ik ], new_peso );
{17} end if;
{18} else
{19} for all j not in [ i1, i2, ....., ik ];
{20} new_peso = peso + A(ik, j);
{21} New_S = ( k + 1, [ i1, i2, ....., ik, j ], new_peso );
{22} Búsqueda_Exhaustiva_Rec ( New_S, Best_S_so_far );
{23} end for;
{24} end if;
{25} return
{26} end de Búsqueda_Exhaustiva_Rec;
```

Búsqueda_Exhaustiva_Rec recursivamente va generando el árbol de búsquedas hasta encontrar el costo mínimo.

Este algoritmo busca todos los (n - 1)! posibles caminos iniciando en 1, guardando el mejor. Hay mucho de paralelismo, porque después de k recursivos llamados a Búsqueda_Exhaustiva_Rec, hay (n - 1)*(n - 2)*.....*(n - k) subárboles independientes para buscar, el cual puede ser realizado en muchos procesadores.

El hecho de que se realice una búsqueda de todos los (n-1)! distintos caminos en el peor de los casos implica que tiene un O(n!).[25]

Así al estudiar el algoritmo podemos indicar de los puntos más importantes que en :

- {02} Calcular el valor inicial del peso, dónde el peso depende del recorrido inicial que puede ser uno escogido arbitrariamente, así que en este método se escogió {1,2,3,4, ...n}, el cálculo del peso matemáticamente se representa como :

$$\text{peso}=A(n,1) + \sum_{i=1}^{m=n-1} A(i,i+1)$$

  se agrega el valor A(n,1) para que pueda cerrarse el circuito y su peso respectivo sea agregado.
- {03} Inicializar el circuito como :

  nodo(i) = i con i = 1,2,3,....., n

  dónde n es el número total de nodos, vértices o de puntos de distribución . El nodo(i) se deja en el registro Best_S_so_Far.
- {04} El registro S contendrá el circuito que se va formando en la recursividad, debe ser inicializado en (1, [1], 0) que significa que es el

[25] Jonson David S. , Aproximation Algortihms for Combinational Problems . Journal of Computer and System Sciencies , Vol 9. pags 256- 278 ,1974

primer nodo, el conjunto que contiene la ruta que se va formando contiene al nodo No. 1 cuyo peso es 0.

- {05} Llamar al procedimiento de Búsqueda_Exhaustiva_Rec para la búsqueda del costo mínimo y la ruta respectiva.
- {09} y {10} Recibir la información que llega a las variables del procedimiento
- {11} Si k = n significa que ya se terminó de generar un circuito parcial, es decir deja de ser circuito parcial para ser circuito total y debe,
- {12} terminar de calcular el peso New_peso, al que se le sumará el peso que se encuentra en la posición A(S,nodo(k),1). Donde k es el último nodo del circuito. La columna debe ser la 1 para cerrar el circuito al nodo 1, porque este fue el inicial.
- {14} a - {17} Guardar en el registro New_S ← S siempre y cuando el peso New_peso recién calculado sea menor que el peso pesoB anexando los datos recién calculados.
- {19}a - {24} Se crean los circuitos con sus respectivos pesos y repetir la operación en forma recursiva.

Posteriormente el método modificado o con "poda" o acotamiento es el siguiente:

```
{01}Procedure Ramificación_Acotamiento_ingenuo_Rec ( S, Best_S_so_far )
{02} Let S = ( k, [ i1, i2, ....., ik ], peso )
{03} Let Best_S_so_far = ( n, [ i1B, i2B, ....., inB ], pesoB )
{04} If k = n
{05} then
{06} new_peso = peso + A(ik, i1)
{07} if new_peso < pesoB
{08} then
{09} Best_S_so_far = ( k, [ i1, i2, ....., ik ], new_peso )
{10} end if
{11} else
{12} for all j not in [ i1, i2, ....., ik ]
{13} new_peso = peso + A(ik, j)
{14} if new_peso < pesoB
{15} then
{16} New_S = ( k + 1, [ i1, i2, ....., ik, j ], new_peso )
{17} Naive_Branch_and_Bound_Search ( New_S, Best_S_so_far )
{18} end if
{19} end for
{20} end if
{21} return
{22} end de Ramificación_Acotamiento_ingenuo_Rec
```

Como puede verse el algoritmo es similar al de Búsqueda Exhaustiva Ingenua, por lo que solo se hará hincapié en las líneas que marcan la diferencia, que son:

{14} a -{18} Se compara si el peso recién calculado new_peso es menor que el pesoB anterior. Si es menor guarda como Best_S_so_Far al calculado, pero si

resulta ser mayor el peso nuevo New_peso entonces ya no llama a la recursividad. ¡He aquí el ahorro!, dado que ya no termina los circuitos parciales que están en estas circunstancias reduciendo las búsquedas de los (n-1)! posibles caminos del método anterior. Pero cabe aclarar que también en el peor de los casos hace la búsqueda de los (n-1)! caminos siendo por ello de O(n!)[26].

Respecto al adaptación prim tenemos:

```
{01} Function Adaptación_Prim
{02} begin
{03} Se inicializa el vector col a ceros;
{04} menor_en_fil ←•;
{05} dir ←0;
{06} i ← 0;
{07} Best_S_so_Far[0] ← 0;
{08} col[0] ← 1;
{09} for j ← 1 to n_vertices - 1 do
{10} begin
{11} menor_en_fil ←•;
{12} for k ←0 to n_vertices -1 do
{13} begin
{14} if (col[k] = 0) and (menor_en_fil > a[Best_S_so_Far[i]][k])
{15} begin
{16} menor_en_fil ← a[Best_S_so_Far[i]][k];
{17} dir ← k;
{18} end del if;
{19} end del for k;
{20} Best_S_so_Far[j] ←dir;
{21} col[Best_S_so_Far[j]] ← 1;
{22} i ← j;
{23} w ← w + menor_en_fil;
{24} end del for j
{25} w ← w + a[Best_S_so_Far[n_vertices-1]][0];
{26} return w;
{27} end de la function Adaptación_Prim
```

Se explicarán las líneas más importantes:

{03} Se lleva un registro para cada vértice *x* en *V(G)* del vértice *u* en *a* con menor valor *a(u,x)* en el vector *col* que es inicializado en ceros. Cuando se localice un vértice adecuado la posición correspondiente se marcará con 1.
{04} y {10} Para localizar el menor de cada fila de la matriz de adyacencia se inicializa el menor_en_fil en •.
{07} Para ir guardando la ruta se usa el vector Best_S_so_Far que es inicializado con el nodo 0 en la dirección 0;
{08} El vector col es marcado con 1 porque ya se tomo el nodo 0 como punto de partida.

---

[26] Articulo CS267. Assignament 5 : Traveling Salesman Problem , Due March 21, 1995

{09} El ciclo for corre desde 1 hasta n_vertices - 1 porque ya se tomo el nodo 0 anteriormente.
{12} a {19} Se busca la arista con menor valor en peso.
{20} a {21} se localiza la dirección de esa arista y se guarda en el vector Best_S_so_Far marcando al vector col con 1 en la dirección de Best_S_so_Far[j].
{23} Se actualiza el peso sumándole a este el menor_en_fil localizado.
{25} Aquí se adapta para que se cierre el camino Hamiltoniano, sumando al peso el peso de la matriz en la dirección de Best_S_so_Far[n_vertices-1]][0].

El tiempo para encontrar la x más cercana en *a* y después actualizar registros es O(n). Pero finalmente en el peor de los casos corre en un tiempo $O(n^2)$ por los dos ciclos anidados.

Asi tenemos nuestros algoritmos para la obtención de la ruta optima , aplicable tanto a cantidades pequeñas como a cantidades grandes.

3. Diagrama Entidad Relación Del Modelo Propuesto.

A continuación se presenta el diagrama entidad relación del modelo propuesto cabe mencionar que se integraron 2 tablas adicionales para el perfil de seguridad del acceso por medio de la aplicación el cual estará a cargo del software PHP Secure Pages que es una herramienta confiable en niveles de seguridad bastante buena.

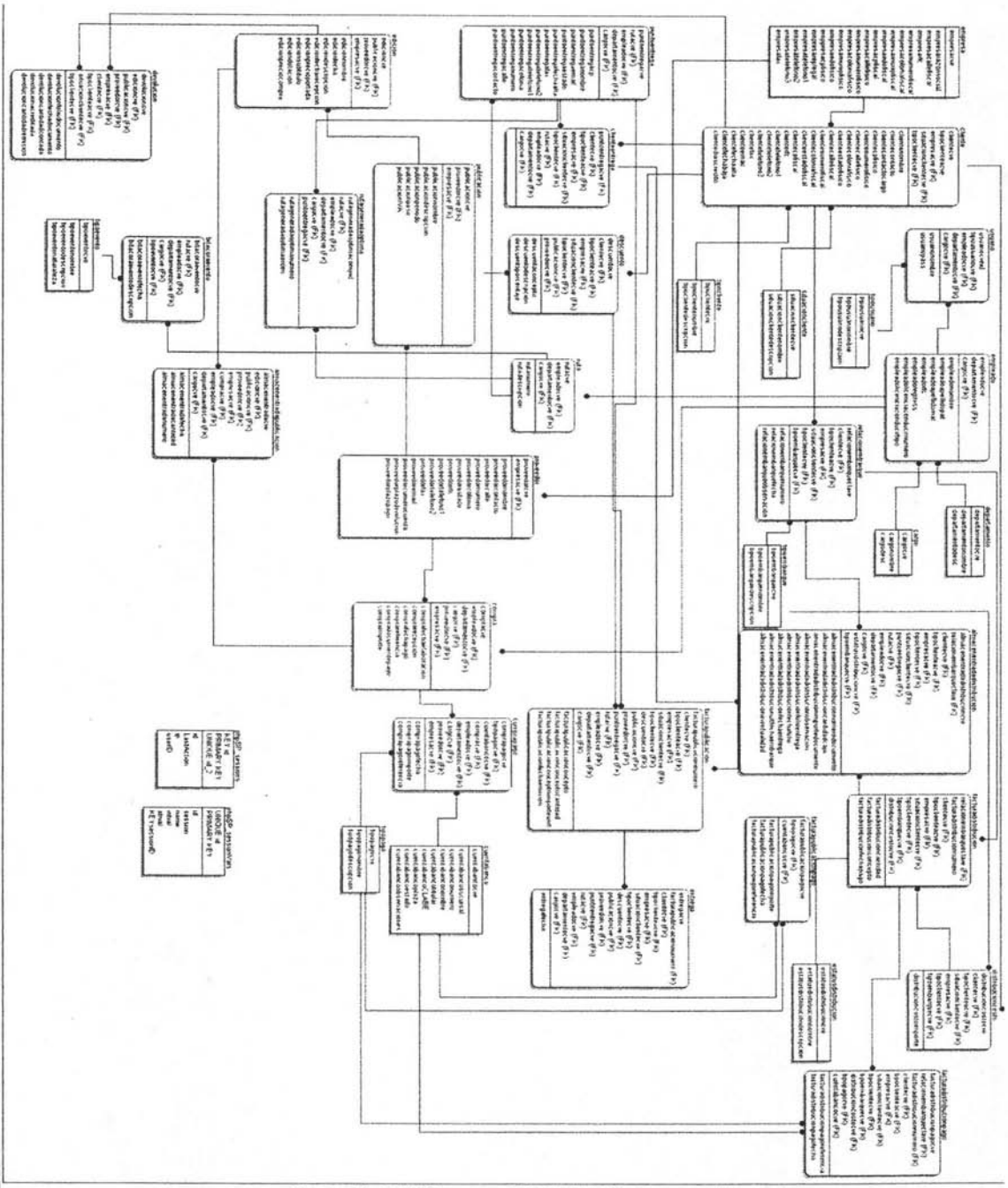


Ilustración 14 " Diagrama ER del Modelo Propuesto"

4. Interfase.

El principal objetivo de la interfase es el de ocultar al usuario del sistema las complejidades del mismo y por el contrario presentarle la información de una manera amena y agradable que le permite integrar el sistema a su forma cotidiana de trabajo. Como se ha mencionado anteriormente los objetivos de nuestro sistema son:

- FUNCIONALIDAD: Que el software haga el trabajo para el que fue creado.
- CONFIABILIDAD: Que lo haga bien.
- DISPONIBILIDAD: Que todos los que quieran utilizar el sistema no tengan problemas.
- SEGURIDAD: La persona que no esté autorizada no debe tener acceso al sistema o parte de él.
- INTEGRIDAD: Que la información sea verídica e igual en todos lados.
- ESTANDARIZACIÓN: Las características de la interfaz de usuario deben ser comunes entre múltiples aplicaciones.
- INTEGRACIÓN: Que todos lo módulos sean fáciles de accesar.
- CONSISTENCIA: Que el apoyo visual sea igual en todas las pantallas (Ej. ventanas similares). También se refiere a la terminología y los comandos usados en la interfaz.
- PORTABILIDAD: Que el paquete sea reconocido por la mayoría de las computadoras.

Para lograr estos objetivos juega un papel importante la interfase ya que en gran medida depende de la aceptación del usuario, de estos usuarios es necesario identificar que tipo de usuario es (no se refiere a que área es, por ejemplo usuario del área contable , usuario de consulta etc. , sino al tipo de conocimiento que de acuerdo a su perfil tiene el mismo) así podemos identificar a los siguientes tipos de usuarios :

- Principiante.-
  - No posee conocimiento sintáctico
  - Debe proporcionársele un número limitado de comandos
  - Debe ofrecerse retroalimentación
  - Los mensajes de error deben ser específicos
  - Herramienta: Tutórales y ayuda en línea
- Intermedio
  - Posee conocimiento semántico
  - Generalmente tiene poco conocimiento sintáctico
  - Se recomienda usar estructura consistente en la interfaz
  - Protección para permitir explotar el sistema
  - Herramienta: ayuda en línea
- Experto
  - Posee alto conocimiento sintáctico y semántico
  - Desean terminar rápido
  - Se recomienda proporcionar macros y short-cuts (atajos)

Como se puede observar es importante identificar el tipo de usuario que interactuara en el sistema , para poder generar también los perfiles de las tareas que se diseñaran en la interfase , entre ellas las que se consideran son:

- Tareas frecuentes
  - Lo conveniente es asignarlas a teclas de función.
  - Hacer íconos o botones para éstas acciones
  - Ejemplos de acciones frecuentes: Guardar, Introducir, Imprimir, etc.
- Acciones intermedias frecuentes
  - Asignarles una combinación de 2 teclas (Ctrl+C, Alt+F10)
  - Deben estar en un menú
  - Ejemplo: Modificación, Seleccionar Todo etc.
- Acciones menos frecuentes
  - Asignarlas a menús anidados
  - Ejecutarlas por medio de líneas de comando
  - Por ejemplo: Corregir, Insertar un elemento en otro.
- Acciones poco frecuentes
  - Colocarlas en menús, submenús, subsubmenús, etc.
  - Usar formas
  - Por ejemplo: Cambiar el protocolo de red, configurar impresoras.

Mas aun resulta interesante como se interrelacionan otras áreas que inicialmente se pensaba totalmente ajenas a nuestra área de estudio como lo es la Psicología, ya que al diseñar nuestra interfase, es importante considerar como razonara el usuario y de tal manera hacer la interfase de acuerdo a esa forma de pensar, y que lleve la secuencia acorde a ella, en este apartado existen 3 teorías:

4.1. Teoría De Los Cuatro Niveles. En esta teoría la persona razona en cuatro niveles, es decir, tienen cuatro niveles de abstracción: Estos niveles de abstracción son:

- Léxico.- Es el lenguaje que hablamos. Ejemplo: A, B, C, etc. así podemos mencionar que aprendemos que el símbolo A es una "A"
- Sintáctico.- Combinar para hacer palabras. Ejemplo: casa, Avión, etc. Combinamos letras para hacer palabras.
- Semántico.- Es el significado de las palabras. Ejemplo: casa.- Lugar donde viven las personas.
- Conceptual.- Es la idea, lo que tengo en mente. Ejemplo: Esa cAsa está muy bonita.

En nivel de interfaces lo podemos ver de la siguiente forma:

- Conceptual.- Es la idea del sistema como un todo (En nuestro caso el sistema de gestión y de operación pero puede ser una hoja de cálculo, calculadora, etc.)
- Semántico.- Es el significado de: Pantallas e Instrucciones.
- Sintáctico.- Es la unidad de: Comandos y Secuencias.
- Léxico.- Son las dependencias con la computadora

4.2. Teoría G. O. M. S. (Globals Operators Methods Selection). Afirma que un individuo se fija metas que se pueden dividir en submetas y así sucesivamente. Cada submeta nos lleva a un diferente objetivo.

Para llevar a cabo estas metas hay:

- Operadores.- Son entidades finitas
- Métodos.- Son acciones
- Reglas de selección.- Sirven para escoger los métodos adecuados

Un ejemplo de este modelo es la acción de Mover Texto. Mover el texto es la meta principal, que se puede conseguir a través de varias submetas: marcar el texto, cortar el texto, pegar el texto.

4.3. Teoría De Las Siete Etapas. Cuando el hombre interactúa con la computadora, se forma una meta. Esta meta se puede lograr realizando o llevando a cabo siete pasos básicos.

MODULO DE INICIO

- Formar la meta.
- Formar la intención.

MODULO DE EJECUCIÓN

- Especificar la acción.
- Ejecutar la acción.
- Percibir el estado del sistema.
- Interpretar el resultado

BRECHA DE EVALUACIÓN

- Evaluar el resultado.

MODULO DE INICIO: El usuario se forma en la mente la meta y la intención que desea sobre el sistema.

MODULO DE EJECUCIÓN: Diferencia entre las acciones que se quieren hacer y las que se pueden hacer en el sistema.

MODULO DE EVALUACIÓN: Diferencia entre el estado resultante del sistema y el estado que se esperaba

Un ejemplo de esta teoría puede ser el siguiente:
Quiero dar de alta un punto de distribución *(meta)*
Ya lo voy a dar de alta *(intención)*
¿Cómo lo doy de alta? *(especificar acción)*
Dar de alta *(ejecutar la acción)*
Corroborar si lo di de alta *(percibir el estado del sistema)*
¿El resultado es el esperado? *(interpretación)*
Decido si está correcto según mis expectativas *(evaluación)*

Por otra parte al momento de realizar la planeacion del diseño es necesario considerar los factores humanos como el porcentaje de errores que comete el usuario, la satisfacción del usuario para realizar determinadas tareas etc. Así como la adecuación de la misma para diferentes usuarios de esta manera

podemos emplear CSS para definir posibles combinaciones de colores. Como se muestra a continuación:

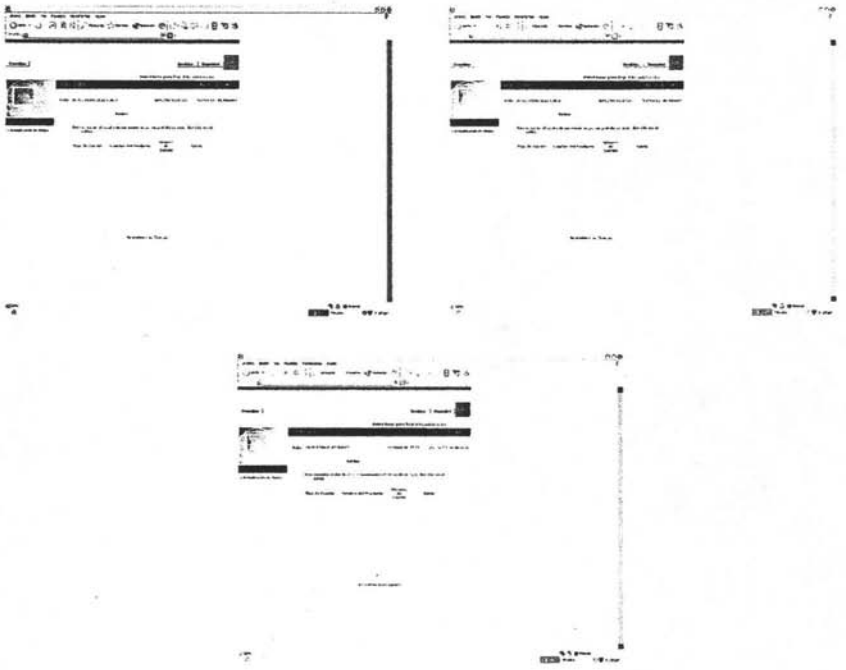


Ilustración 15 "Diferentes Combinaciones de colores en la interfase"

Respecto a la interacción de la interfase con el usuario podemos mencionar 8 reglas para el diseño de diálogos , los cuales son:

- Buscar la consistencia.
- Permitir el uso de atajos.
- Ofrecer retroalimentación informativa.
- Diseñar diálogos que obliguen a generar secuencias completas.
- Ofrecer manejo simple de errores.
- Permitir deshacer acciones fácilmente.
- Motivar la sensación de control.
- Reducir la carga de memoria a largo plazo.

Y finalmente los puntos a considerar son los de la captura de datos y el despliegue de ellos. Los cuales se enlista a continuación:

4.4. Guía Para Despliegue De Datos.

- ORGANIZACIÓN
  i. Consistencia.- Que todos los datos sean iguales, formatos, etc.
  ii. Asimilación eficiente de información.- Estructura de la información y buena redacción. Alineado, etc.
  iii. Carga de memoria mínima para el usuario.
  iv. Compatibilidad entre el despliegue y la entrada de datos.
  v. Flexibilidad para el control del despliegue.- Ordenar por nombre, por tipo, etc.
- ATRAER LA ATENCIÓN DEL USUARIO
  i. Intensidad (sólo dos niveles)
  ii. Resaltado (Subrayado, encerrado, viñetado)
  iii. Tamaño (lo recomendado es no más de 4 tamaños)
  iv. Tipos (lo recomendado es no más de 3 tipos)
  v. Video inverso
  vi. Parpadeo
  vii. Color (lo recomendado es no más de 4 colores)
  viii. Cambio de color
  ix. Audio (tonos suaves para retroalimentación positiva, tonos ásperos para emergencias).

4.5. Guía Para Captura De Datos.

- Consistencia en las transacciones de entrada
- Acciones de entrada mínima
- Carga de memoria mínima
- Compatibilidad entre captura y despliegue de datos
- Flexibilidad en el control de captura

Se observaran y cuidaran durante el desarrollo de los módulos todos estos factores para obtener un sistema con una interfase eficiente y agradable.

CUARTO CAPITULO "DISEÑO E IMPLEMENTACIÓN DEL SISTEMA Web."

1. Puntos Estratégicos De La implementación.

Una vez finalizado el respectivo análisis completo de nuestro sistema la tarea siguiente será la del diseño y la implementación, en este apartado tendremos que observar puntos estratégicos tales como el tiempo estimado de desarrollo , planeación de tareas así como la forma en cómo se realizara dicha implementación la que puede ser por fases , por módulos o por funciones del mismo , para esto es necesario considerar las necesidades de la empresa en dónde se implementara así como los requerimientos de la misma para sus funciones cotidianas.

Sin embargo los puntos principales de las actividades de la construcción e implementación del sistema son:

1.1. Desarrollo De Componentes Del Sistema.

El objetivo principal de este punto es el de preparar todos los componentes necesarios para el correcto funcionamiento del sistema
Preparar el entorno de desarrollo, pruebas y procedimientos para el mismo.
Actividades y Tareas Técnicas

➢ Selección de Hardware necesario. ➢ Instalar el hardware necesario. ➢ Selección de Software "plataforma " ó SO para el sistema. ➢ Instalación de Software plataforma ó SO para el sistema. ➢ Prueba de SO con Hardware ➢ Formación de Equipo para el Desarrollo. ➢ Implementar el Sistema Controlador de la Base de datos física para el sistema.	Análisis Individual.
➢ Preparación de entornos de Prueba. o Creación de entornos de prueba (Bases de datos de Prueba. o Definir estrategias de prueba.	Prueba y análisis Individual.

○ Definir datos comunes de prueba. ○ Definir procedimientos, operaciones y estándares de desarrollo.	
➢ Diseño detallado de módulos. ➢ Programación y prueba individual. ➢ Evaluación de las pruebas unitarias. ➢ Documentación de los módulos y componentes del sistema.	Prueba individual. Diseño y programación
➢ Identificar y evaluar posibles modificaciones al análisis. ➢ Realizar Pruebas de integridad al sistema.	Análisis Prueba

Tabla 9 "Plan del desarrollo del sistema"

1.2. Desarrollo De Procedimientos Y Manuales De Usuario.

Procedimientos y manuales de usuario.	
Actividades y Tareas	Técnicas
➢ Desarrollar plan de capacitación de los usuarios, ○ Identificar perfiles de usuario. ○ Identificar necesidades y recursos para la capacitación de los usuarios. ○ Preparar los materiales de capacitación para los usuarios.. ○ Planificar el tiempo y la cantidad de integrantes del grupo de capacitación.	Programación Análisis
➢ Desarrollar manuales de usuario. ➢ Revisión de los manuales de usuario. ➢ Entrega de manuales de usuario.	Diseño.

Tabla 10 "Desarrollo de Procedimientos y Manuales de Usuaurio."

1.3. Migración De Datos.

En el caso de contar previamente con un DBMS[27] o sistema, es necesario realizar la migración de datos al nuevo sistema, para ello también es importante contar con un plan de migración para evitar posibles problemas que podrían ser de un riesgo muy grande como el no poder operar la organización de forma normal o simplemente el no ser posible su funcionamiento.

Este plan de tareas se prepara cuando el nuevo sistema está completamente desarrollado y se han realizado las pruebas de integridad.

Plan de Migración de Datos.

Actividades y Tareas	Técnicas
➢ Identificar datos a migrar. ➢ Desarrollar software específico para la migración o en su caso las consultas necesarias al DBMS. ➢ Desarrollo de Procedimientos específicos de migración. ➢ Captura y preparación de datos.	Análisis
➢ Realizar la migración de datos. o Realizar copia de seguridad de los datos. o Llevar a cabo la migración. o Revisión post migración.	Programación .

Tabla 11 "Plan para la migración de datos"

En el caso particular de la organización no se cuenta con datos para migrar.

1.4. Revisión Del Plan De Instalación.

El plan de instalación se desarrolla al finalizar las tareas del diseño de nuestro sistema. A partir de esto se establecerán los recursos necesarios y se planificaran en el tiempo teniendo en cuenta su disponibilidad.

[27] DBMS "Data Base Manager System" o " Sistema Manejador de la Base de Datos".

Desarrollar el plan de instalación

Actividades y Tareas	Técnicas
➢ Estimar los recursos o Equipo de Trabajo. o Software y Hardware. o Tiempos y Costos.	Estimación.
➢ Planificar o Actividades. o Tareas. o Recursos.	Planificación.

Tabla "Plan de Instalación del Sistema"

1.5. Inicio De Operación Del Sistema.

Esta actividad implica la entrega del producto final a los usuarios para que estos lo utilicen de forma completa. Previamente el software debió de haber sido probado , todos los datos globales del sistema definidos , la migración de datos se debió de haber realizado y los usuarios ya deberán de encontrarse capacitados para su uso.

Para el inicio de operación del sistema existen varias estrategias de implementación, la elección de una u otra dependerá del sistema mismo, de la organización para la cual se esta implementando, del numero de áreas y el número de usuarios involucrados.

Estrategia	Desventajas	Ventajas
Sistemas Paralelos.	Costo Alto en tiempo y Horas Hombre. Resistencia por parte del usuario a la incorporación del sistema por la cultura del uso del anterior.	Mayor seguridad ante posibles contratiempos en el inicio de operaciones.
Conversión Directa	Los problemas se producen sobre los procesos y operaciones de la empresa.	Inicio de operaciones sin duplicidad.
Enfoque Piloto	Los errores del sistema producen perdida de confianza en las demás áreas de la empresa.	Los problemas afectan a pequeñas áreas , ó áreas de prueba y pueden solucionarse antes de la implementación total ó definitiva
Por Etapas	Largos periodos de migración e implementación. Perdida de confianza al tiempo transcurrido.	No se ven ventajas significativas sólo en caso de éxitos unitarios en el proceso de inicio de operaciones.

Tabla 12 "Estrategias de Inicio de Operación del Sistema"

Cabe mencionar que es muy importante verificar y comprobar el funcionamiento correcto del sistema antes de su inicio de operaciones independientemente de la estrategia usada.

1.6. Tabla de Tiempos.

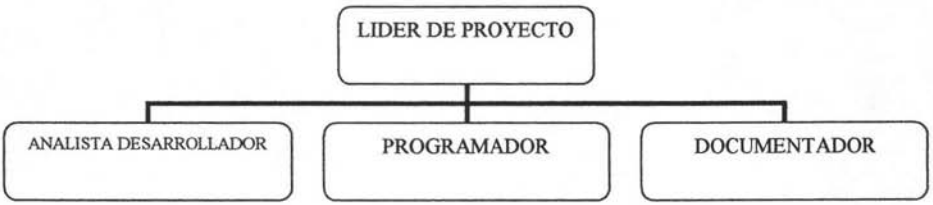
Es muy recomendable realizar para las numerosas tareas, de ésta fase la tabla de tiempos respectiva a cada tarea lo que es una estimación del tiempo de las mismas, de esta manera tendremos metas a corto plazo para ir completando y no prolongarse demasiado del tiempo estimado para el desarrollo de la misma.

Un ejemplo de tabla de tiempo es el siguiente:

Id	Nombre de la Tarea	T1			T2			T3			T4			T1		
		ENE	FEB	MAR	ABR	MAY	JUN	JUL	AGO	SEP	OCT	NOV	DIC	ENE	FEB	MAR
1	Implementacion del Sistema.															
2	Analisis y Adquisicion de Hardware Necesario.															
3	Instalacion de HW y SW Necesario para el sistema.															
4	Capacitacion de Usuarios.															
5	Recopilacion de errores presentados del sistema.															
6	Correccion de Errores y puesta a punto.															
7	Revision post correccion.															
8	Analisis de cambios y modificaciones a mediano y largo plazo.															
9	Mantenimiento del Sistema.															

2.- Equipo De Trabajo Para El Desarrollo E Implementación.

El equipo de trabajo para el desarrollo del sistema estará constituido por el siguiente esquema:



Este equipo se ha considerado en base a los tiempos ponderados para su implementación y al tamaño del proyecto.

Estos miembros del equipo integraran totalmente las diferentes tareas de la fase en un tiempo razonable, dada la complejidad del mismo.

QUINTO CAPITULO "PERSPECTIVAS DE DESARROLLO."

1. Generalidades.

El presente proyecto presenta una fuente muy importante para el crecimiento de la organización para el cual esta orientado, no sólo por la competitividad tecnológica que implica sino por toda la información para la toma de decisiones y el análisis de resultados que se generaran a partir de él así como la implantación de procedimientos y políticas administrativas bien definidas.

El alcance del proyecto ha sido cubierto satisfactoriamente, el cual como perspectiva de desarrollo tiene la concertación de su totalidad de los diferentes módulos al ser un sistema a una escala considerablemente complejo así como de la realización de las pruebas pertinentes necesarias para su validación y puesta a punto.

Más aun se abre la posibilidad de no solo contemplar el sistema Web a nivel Intranet sino a nivel Internet, lo que permitiría la colaboración de las diferentes áreas de trabajo y sus integrantes a distancia, permitiendo la flexibilidad de crecimiento sin la necesidad de un costo alto así como la integración de servicios informáticos eliminando la duplicidad de funciones y redundancia de información en las áreas de la organización. Por otra parte, fue diseñado para una posible diversificación de las actividades de la empresa, esto es, permite la integración de diferentes tipos de "empresas" que integren a la organización afectando a su totalidad los resultados de la misma, así, no se tendrá que trabajar con determinada empresa, determinado sistema sino solo 1 que proporcionara toda la información de análisis, contable, toma de decisiones y control de la organización.

Posteriormente se podrá hacer un análisis más a detalle del mejoramiento del algoritmo para la optimización, en función del comportamiento del resultado de este así como su perfeccionamiento a corto plazo.

Por otra parte no solo se ha proporcionado a la organización de información valiosa para su consolidación formal sino que la recopilación de información ha dado pauta para generar los diferentes manuales de procedimientos y operativos lo que consolida la organización para ser más estructurada.

Queda abierto el apartado del desarrollo de los diferentes módulos administrativos que se han contemplado, no solo por el tamaño de los mismos sino también para su validación y comprobación del funcionamiento de ellos, sólo se tendrán que hacer a nivel de interfase ya que a nivel de datos ya estan contemplados como se ha observado en el diagrama ER.

Finalmente cabe mencionar que con toda la documentación realizada del sistema cualquier persona capacitada en el área podrá darle mantenimiento y cambios sin ningún problema evitando la dependencia de la organización sobre organismos externos y / o terceras personas, redundando en un ahorro de recursos para ella.

2. Puntos de Mejora y Extensión.

A corto plazo se tendrá que definir el costo relativo de cada punto de distribución en función de las consideraciones pertinentes como lo es el gasto de combustible , el kilometraje recorrido , el plan de eventos presentado etc. , y no solo se podrá utilizar para optimizar rutas ya que el modelo de datos alimentara a la gerencia o dirección información para la toma de decisiones como análisis de costos contra ventas , consolidación de carga en almacén , gastos operativos , eficiencia de operación , estadística informativa de eventos y procesos en un intervalo de tiempo determinado , prácticamente los resultados que puede arrojar el sistema son extremadamente completos , de aquí partimos de la necesidad de mejora continua de la misma para la implementación de dichos módulos.

Ademas se tiene la opcion de llevar el sistema con salida a internet con lo que permitira una flexibilidad muy amplia para el trabajo a distancia y sus diversas aplicaciones tales como comercio electronico o asignación de apartados para proveedores y clientes. Todo ello ha sido contemplado en el analisis y de esta forma lo tenemos preparado para todas las mejoras que se deseen implementar.

Respecto a la optimizacion de la ruta de distribución se podra considerar una restricción de acuerdo a la distancia para recorrer en funcion de la capacidad de la unidad. Mas sin embargo no se considera critico en este momento, pero si se puede implementar.

CONCLUSIONES.

1. Generalidades.

Finalmente después del estudio de los diversas notaciones de desarrollo para la implementación de sistemas Web se han alcanzado los objetivos al tener conocimientos sólidos en el diseño de bases de datos así como su implementación de aplicaciones (interfases) que pueden ser accesadas por medio de servidores Web tal es el caso del servidor Apache. Durante el desarrollo del presente han surgido conocimientos que son fundamentales para un diseño exitoso y la implementación de sistemas eficientes como los diversos puntos en el diseño de bases de datos , de interfases así como la cualidad que debe de tener un Analista de Sistemas y no sólo comportarse como un programador , que tenga conocimientos de lenguaje y sólo se dedique a "tirar" código , por el contrario se tiene que tener en mente una estructura ordenada para crear librerías que sean fácilmente mantenibles a largo plazo y en caso de cambios no se tenga que remplazar gran parte del código.

Resultan peculiarmente interesantes los diversos puntos que se necesitan cuidar durante el desarrollo de la interfase ya que es muy importante tener una que sea extremadamente amigable y funcional, en donde el usuario se sienta cómodo y acepte la integración con el sistema.

Para la solución de la optimización de rutas se estudiaron diferentes métodos exactos como aproximados, seleccionando para su implementación el de "Búsqueda Exhaustiva Ingenua (Naive Search Exhaustive)" y se contemplo la opción de implementar el de "Ramificación y Acotamiento Ingenuo (Naive Branch and Bound)" para su posible mejora en cantidades muy grandes de datos (no mas de 9) y el metodo adaptación prim para cantidades superiores. De esta forma tenemos un sistema óptimo con flexibilidad de funcionamiento.

2. Resultados.

El objetivo general que era el de proporcionar el análisis, diseño y las bases de la implementación del proyecto con los algoritmos de IA. para la organización se satisface al haber realizado dichos puntos en el presente , quedando abierta la conclusión de la implementación por la magnitud del proyecto que es de una complejidad muy considerable además de que con el diseño de los casos de uso en la herramienta Racional Rose y el diagrama Entidad Relación en la herramienta Erwin prácticamente ya se encuentra diseñada en su totalidad la implementación de la base de datos física ya que permite directamente la generación de código para su implementación. Además se ha generado una cantidad significativa muy importante de procedimientos y políticas administrativas para la organización así como se ha proporcionado competitividad tecnológica a dicha organización con herramientas libres de alta calidad para tener un crecimiento sostenido y enfrentar la competencia con gran eficiencia, dando así solución a sus diversos problemas que presentaba sin una inversión de recursos considerable, por el contrario de muy bajo costo.

3. Aportaciones

La aportación del presente documento consiste en proporcionar un sistema de calidad, eficiente y bien documentado a la organización en estudio permitiéndole un crecimiento a corto y largo plazo así como la integración de algoritmos para optimizar su operación la cual consiste en la generación de rutas de distribución de manera óptima. A la vez de ser un sistema portable y accesible por medio de tecnología Web. No solo se provee la solucion sino tambien una fuente de consulta para el area de optimizacion de operaciones aplicable a otras areas asi como su estudio directamente a la optimizacion de rutas para distribucion. Finalmente cabe mencionar que ademas se han proporcionado una serie de politicas y procedimientos bien definidos que permite una mejor administración a la organización la cual carecia de los mismos.

4. Limitaciones.

La limitación principal fue la del desarrollo de la aplicación en su totalidad ya que esta misma es de una gran complejidad y se requiere de mucho más tiempo así como de la integración de equipos de desarrollo para obtener un producto de calidad como se ha planteado en el análisis. Y por tratarse de un sistema Web no presenta limitaciones respecto a hardware salvo que tenga la posibilidad de usar un explorador web. Respecto a la optimizacion de rutas se presenta la opcion de muchos a pocos puntos de distribución siendo muy flexible respecto a este apartado.

BIBLIOGRAFÍA.

- Angel Reyes González : Tesis Un ejemplo de Optimización de Funciones por Medio del Algoritmo Ariad's Clew : El caso del Problema del Agente Viajero. Universidad de las Américas Puebla; Departamento de Ingeniería en Sistemas Computacionales. 1997.
- A.V. Aho, J.E. Hopcroft y J.D. Ullman: The Design and Analysis of Computer Algorithms. Addison Wesley. 1974.
- Connolly, T.; Begg C.; Strachan, A. Database Systems: A Practical Approach to Design, Implementation and Management. 2nd edition. Addison-Wesley.
- D. R. Stinson; An Introduction to the Design and Analysis of Algorithms; Second Edition (Revised); Winnipeg, Manitoba, Canada. 1987.
- De Miguel, A.; Piattini, M.; Marcos, E. Diseño de bases de datos Relacionales. Ra-Ma.
- De Miguel, A.; Piattini, M.: Concepción y diseño de bases de datos: del Modelo E/R al Modelo Relacional. Ra-Ma.
- E. Horowitz y S. Sahni: Fundamentals of Computer Algorithms. Springer Verlag. 1978.
- E.L., Lenstra J.K., Rinnoo y Kan A.H.G., Shmoys D.B.,The Traveling Salesman Problem : A Guided Tour of Combinatorial Optimization, A Wiley - Interscience Publication. 1985.
- Edward M. Reingold, Jurg Nievergelt, Narsingh Deo; Combinatorial Algorithms Theory and Practice; Prentice-Hall, Inc., Englewood Cliffs, New Jersey 07632. 1977.
- Elmasri, R.; Navathe, S.B.: Sistemas de bases de datos. Conceptos fundamentales. 2ª Edición. Addison-Wesley Iberoamericana.
- G.S. Maddala. Introducción a la Econometría. Prentice-Hall Hispanoamericana, S.A. 1996.
- H. S. Wilf: Algorithms and Complexity. Prentice Hall. 1986.
- Harvey Gerber; Algebra lineal; Grupo Editorial Iberoamérica. 1990.
- Hopfield, J. and Tank, D. (1985). Neural computation of decisions in optimization problems. Biological Cybernetics, 52:141 - 152. 1985.
- J. A. Bondy and U.S.R. Murty, Graph Theory with Applications, North-Holland New York.Amsterdam.Oxford. 1976.
- James A. McHugh, Algorithmic Graph Theory, Prentice Hall, Englewood Cliffs, New Jersey 07632. 1990.
- Jeffrey D. Smith, Design and Analysis of Algorithms, PWS-Kent Publishing Company. 1989.

- Johnson D. S. Aproximation Algorithms for Combinatorial Problems. Journal of Computer and Systems Sciences. Volumen 9, pp. 256-278. 1974.
- JuanManuel Ahuactzin and Kamal K. Gupta. The Kinematic Roadmap: A Motion Planning Based Global Aproach for Inverse Kinematics of Redundant Robots. IEEE Transactions on Robotics and Automation, Vol. 15, No. 4, August 1999.

- Kurt Mehlhorn. Data Structures and Algorithms 2: Graph Algorithms and NP-Completeness. Springer-Verlag Berlin Heidelberg New York Tokyo. 1984.
- Marvin C. Paull. Algorithm Design A recursion Transformation Framework, Rutgers University. A Wiley-Interscience Publication, John Wiley & Sons. 1988.
- Maty E.S. Loomis, Estructura de Datos y Organización de Archivos pHH Prentice Hall. 1985.
- Mendenhall, Scheaffer, Wackerly. Estadística Matemática con Aplicaciones. Grupo Editorial Iberoamérica. 1986.
- Mehdi Behzad,Gary Chartrand y Linda Lesniak-Foster; Graphs & Digrafos; Wadsworth International Group. 1981.
- Michael. R. Garey / David.S. Johnson: Computers and Intractability A Guide to the theory of NP-Completeness ; W.H. Freeman and Co., New York, 1979.
- Morrow M., Genetic Algorithms a new class of searching algorithms, Dr. Dobb's Journal. Abril 1991.
- Oldham, W.B. Neural Networks. Notes. Texas Tech University. USA 1990.
- Richard L. Burden y J. Douglas Faires. Análisis Numérico. 6a. Edición. Matemáticas International Thomson. 1998.
- Rumelhart, David E. ; James L. McClelland and the PDP Research Group. Parallel Distributed Processing. Volume I and II. MIT PRESS. 1986.
- Stephen B. Maurer and Anthony Ralston, Discrete Algorithmic Mathematics, Addison-Wesley Publishing Company. 1991.
- Unibertsitatea. 1999.Elmasri, R.; Navathe, S.B. Fundamentos de Sistemas de Bases deDatos. 3ª Edición. Addison-Wesley.
- Varela Hernández Ma. Del Rosario : Tesis Diseño de una Heurística para el tratamiento del Problema del Agente Viajero; UDLA1996.
- Voigt B. F. Der Handlungsreisende, wie er sein soll und was erzu thun hat, um Auftrage zu erhalten und eins gluchlichen Erfolgs in seinen Geschaften gewiss zu sein, Von eimen alten Commis Voyaguer, Ilmenau, 1831. Y Muller-Merbach. Zweimal travelling Salesman. DGOR-Bulletin 25, 12-13.1983.
- Wasserman, Philip D. Neural Computing. Theory and Practice. Van Nostrand Reinhold. New York. 1989.
- William Anthony Granville.Longley. Calculo Diferencial e Integral, 1992. Pág. 508.
- William W. Hines y Douglas C. Montgomery. Probabilidad y Estadística para Ingeniería y Administración. Tercera Edición. Compañía Editorial Continental, S.A. de C.V. México. 1997.

- Yannakakis Mihalis, On the Approximation of Maximum Satisfiability, The Journal of Algorithms 17, 475-502. 1994.
- Zurada, Jacek M. Introduction to Artificial Neural Systems. West Publishing Company. 1992.

Referencias Web

- http://decsai.ugr.es/~castro/CA/node24.html Complejidad Algorítmica Juan Luis Castro Peña. Depto. Ciencias de la Computación e Inteligencia Artificial. Universidad de Granada. 1999.
- http://journey.cem.itesm.mx/compresiondemo.html Algoritmos Genéticos para Compresión Fractal de Imágenes, Isaac Rudomín, PhD., M.C. Lucía Vences. Mayo de 1995
- http://www.cimat.mx/~horebeek/cursus/node35.html Algoritmos Genéticos. Juan Carlos Morales. Universidad de los Andes , Departamento de Ingeniería Industrial; Cll 1A No. 18A - 70 Bogotá, Colombia Suramérica. 1999.
- http://www.iwr.uniheidelberg.de/iwr/comopt/soft/TSPLIB95/TSPLIB.html Variaciones del Problema del Agente Viajero. Pablo Moscato's Home Page. Departamento de Electrotecnia, Facultad de Ingenieria, Universidad Nacional de La Plata. Argentina. 1999.
- http://www.ing.unlp.edu.ar/cetad/mos/TSPBIB_home.html. 1999. Burr,D.J., An Improved Elastic Net Method for the Traveling Salesman Problem. IEEE Conf. On Neural Networks, San Diego (1988), 69-76.
- http://www.geocities.com/CapeCanaveral/9802/3d5ca000.htm Algoritmos Genéticos P. Larrañaga Departamento de Ciencias de la Computación e Inteligencia Artificial Universidad del País Vasco – Euskal Erico

ANEXO.

1. Listado General De Atributos Del Modelo Entidad Relación.

Atributo	
Nombre	**Restricciones**
almacenentradacantidad	No
Almacenentradacve	No
almacenentradadistribucioncant	No
almacenentradadistribucioncve	No
almacenentradadistribucioneven	No
almacenentradadistribucionfech	No
almacenentradadistribucionfech	No
almacenentradadistribucionfoli	No
almacenentradadistribucionnume	No
almacenentradadistribucionobse	No
almacenentradadistribucionultf	No
almacenentradafecha	No
almacenentradanumero	No
amacenentradadistribucionimpor	No
bitacoraeventocve	No
bitacoraeventodescripcion	No
bitacoraeventofecha	No
cargocve	No
cargocve	No
cargocve	No
cargocve	No
cargocve	No
cargocve	No
cargocve	No
cargocve	No
cargocve	No
cargocve	No
cargocve	No
cargocve	No
cargocve	No
cargocve	No
cargocve	No
cargodesc	No
cargonombre	No
clientedifesacallefiscal	No
clientedifesacallefisico	No
clientedifesacoloniafiscal	No
clientedifesacoloniafisico	No
clientedifesacontacto	No
clientedifesacontactocargo	No
clientedifesacpfiscal	No
clientedifesacpfisico	No
clientedifesacve	No
clientedifesacve	No
clientedifesacve	No
clientedifesacve	No
clientedifesacve	No
clientedifesacve	No
clientedifesacve	No
clientedifesacve	No
clientedifesacve	No
clientedifesacve	No
clientedifesacve	No

Atributo	
Nombre	**Restricciones**
clientedifesadiascredito	No
clientedifesaemail	No
clientedifesaestadofiscal	No
clientedifesaestadofisico	No
clientedifesafax	No
clientedifesafechaalta	No
clientedifesafechabaja	No
clientedifesanombre	No
clientedifesanumerofiscal	No
clientedifesanumerofisico	No
clientedifesarfc	No
clientedifesatelefono1	No
clientedifesatelefono2	No
clientedifesatelefono3	No
compracve	No
compracve	No
compracve	No
compradocumentopago	No
comprafechaelaboracion	No
comprafechapago	No
compraimporte	No
comprapagocve	No
comprapagofecha	No
comprapagoimporte	No
comprapagoreferencia	No
comprarecepcion	No
comprareferencia	No
cuentabancoCLABE	No
cuentabancocve	No
cuentabancocve	No
cuentabancocve	No
cuentabancocve	No
cuentabancoestado	No
cuentabanconombre	No
cuentabanconumero	No
cuentabancoobservaciones	No
cuentabancoplaza	No
cuentabancosucursal	No
cuentabancotitular	No
departamentocve	No
departamentocve	No
departamentocve	No
departamentocve	No
departamentocve	No
departamentocve	No
departamentocve	No
departamentocve	No
departamentocve	No
departamentocve	No
departamentocve	No
departamentocve	No
departamentocve	No
departamentocve	No
departamentocve	No
departamentodesc	No
departamentonombre	No
descuentoconcepto	No
descuentocve	No
descuentocve	No
descuentocve	No
descuentodescripcion	No
descuentoporcentaje	No
devolucionacreditada	No
devolucioncantidadcontada	No

Atributo	
Nombre	**Restricciones**
devolucioncantidadremision	No
devolucioncve	No
devolucionfechadocumento	No
devolucionfoliodocumento	No
distribucioncostocve	No
distribucioncostocve	No
distribucioncostocve	No
distribucioncostoimporte	No
edicioncve	No
edicioncve	No
edicioncve	No
ediciondescripcion	No
ediciondotacion	No
edicionfecha	No
edicionfecharecepcion	No
edicionnombre	No
edicionpreciocompra	No
edicionprecioportada	No
edicionsubtitulo	No
empleadoapellidomat	No
empleadoapellidopat	No
empleadocve	No
empleadocve	No
empleadocve	No
empleadocve	No
empleadocve	No
empleadocve	No
empleadocve	No
empleadocve	No
empleadocve	No
empleadocve	No
empleadocve	No
empleadocve	No
empleadocve	No
empleadolicenciaconducirnumero	No
empleadolicenciaconducirtipo	No
empleadonombre	No
empleadoregimss	No
empleadorfc	No
emppresacpfisico	No
empresacallefiscal	No
empresacallefisico	No
empresacoloniafiscal	No
empresacoloniafisico	No
empresacpfiscal	No
empresacve	No
empresacve	No
empresacve	No
empresacve	No
empresacve	No
empresacve	No
empresacve	No
empresacve	No
empresacve	No
empresacve	No
empresacve	No
empresacve	No
empresacve	No
empresacve	No
empresacve	No
empresacve	No
empresacve	No
empresacve	No
empresacve	No
empresacve	No

Atributo	
Nombre	**Restricciones**
empresaedofiscal	No
empresaedofisico	No
empresafax	No
empresafechaalta	No
empresafechabaja	No
empresamail	No
empresamunipfiscal	No
empresamunipfisico	No
empresanumerofiscal	No
empresanumerofisico	No
empresarazonsocial	No
empresareplegal	No
empresarfc	No
empresatelefono1	No
empresatelefono2	No
empresatelefono3	No
entregacve	No
entregafecha	No
entregafolio	No
estatusdistribucioncve	No
estatusdistribucioncve	No
estatusdistribuciondescripcion	No
estatusdistribucionnombre	No
facturadistribucioncantidad	No
facturadistribucionconcepto	No
facturadistribucionfechapago	No
facturadistribucionnumero	No
facturadistribucionnumero	No
facturadistribucionpagocve	No
facturadistribucionpagofecha	No
facturadistribucionpagoreferen	No
facturapublicacionconcepto	No
facturapublicacionconceptocant	No
facturapublicacionconceptoimpo	No
facturapublicacionfechaemision	No
facturapublicacionfechapago	No
facturapublicacionnumero	No
facturapublicacionnumero	No
facturapublicacionpagocve	No
facturapublicacionpagofecha	No
facturapublicacionpagoimporte	No
facturapublicacionpagoreferenc	No
id	No
id	No
intval	No
ip	No
KEY_id	No
KEYsessionID	No
LastAction	No
name	No
PRIMARY_KEY	No
PRIMARY_KEY	No
proveedorcalle	No
proveedorcolonia	No
proveedorcontacto	No
proveedorcumerocuenta	No
proveedorcve	No
proveedorcve	No
proveedorcve	No
proveedorcve	No
proveedorcve	No
proveedorcve	No
proveedorcve	No

Atributo	
Nombre	**Restricciones**
proveedorcve	No
proveedorcve	No
proveedorcve	No
proveedoremail	No
proveedorestado	No
proveedorfax	No
proveedornombre	No
proveedornumero	No
proveedorplazodevolucion	No
proveedorplazopago	No
proveedorrfc	No
proveedortelefono1	No
proveedortelefono2	No
publicacioncve	No
publicacioncve	No
publicacioncve	No
publicacioncve	No
publicacioncve	No
publicacioncve	No
publicacioncve	No
publicaciondescripcion	No
publicacionIVA	No
publicacionnombre	No
publicacionperiodo	No
publicacionpeso	No
puntoentregacalle	No
puntoentregacolonia	No
puntoentregacontacto	No
puntoentregacp	No
puntoentregacve	No
puntoentregacve	No
puntoentregacve	No
puntoentregacve	No
puntoentregacve	No
puntoentregacve	No
puntoentregaemail	No
puntoentregaestado	No
puntoentregafax	No
puntoentregafechaalta	No
puntoentreganombre	No
puntoentreganumero	No
puntoentregatelefono1	No
puntoentregatelefono2	No
relacionembarqueclave	No
relacionembarqueclave	No
relacionembarqueclave	No
relacionembarqueclave	No
relacionembarquefecha	No
relacionembarquenumero	No
relacionembarqueobservacion	No
rutacve	No
rutacve	No
rutacve	No
rutacve	No
rutacve	No
rutacve	No
rutacve	No
rutacve	No
rutadescripcion	No
rutageneradaoptimacompid	No
rutageneradaoptimanumero	No
rutageneradaoptimaorden	No
rutanumero	No

Atributo	
Nombre	**Restricciones**
session	No
situacionclientecve	No
situacionclientecve	No
situacionclientecve	No
situacionclientecve	No
situacionclientecve	No
situacionclientecve	No
situacionclientecve	No
situacionclientecve	No
situacionclientecve	No
situacionclientecve	No
situacionclientecve	No
situacionclientecve	No
situacionclientedescripcion	No
situacionclientenombre	No
strval	No
tipoclientecve	No
tipoclientecve	No
tipoclientecve	No
tipoclientecve	No
tipoclientecve	No
tipoclientecve	No
tipoclientecve	No
tipoclientecve	No
tipoclientecve	No
tipoclientecve	No
tipoclientecve	No
tipoclientecve	No
tipoclientedescripcion	No
tipoclientedifesacve	No
tipoclientedifesacve	No
tipoclientedifesacve	No
tipoclientedifesacve	No
tipoclientedifesacve	No
tipoclientedifesacve	No
tipoclientedifesacve	No
tipoclientedifesacve	No
tipoclientedifesacve	No
tipoclientedifesacve	No
tipoclientenombre	No
tipoembarquecve	No
tipoembarquecve	No
tipoembarquecve	No
tipoembarquecve	No
tipoembarquecve	No
tipoembarquecve	No
tipoembarquedescripcion	No
tipoembarquenombre	No
tipoeventocve	No
tipoeventocve	No
tipoeventodescripcion	No
tipoeventonaturaleza	No
tipoeventonombre	No
tipopagocve	No
tipopagocve	No
tipopagocve	No
tipopagocve	No
tipopagodescripcion	No
tipopagonombre	No
tipousuariocve	No
tipousuariocve	No
tipousuariodescripcion	No

Atributo	
Nombre	**Restricciones**
tipousuarionombre	No
UNIQUE_id	No
UNIQUE_id_2	No
userID	No
usuariocveid	No
usuarionombre	No
usuariopass	No

2. Listado General De Entidades Del Modelo Entidad Relación.

Entidad
Nombre
almacenentradadistribucion
almacenentradapublicacion
bitacoraevento
cargo
cliente
clienteentrega
compra
comprapago
cuentabanco
departamento
descuento
devolucion
distribucioncosto
edicion
empleado
empresa
entrega
estatusdistribucion
facturadistribucion
facturadistribucionpago
facturapublicacion
facturapublicacionpago
phpSP_sessions
phpSP_sessionVars
proveedor
publicacion
puntoentrega
relacionembarque
ruta
rutageneradaoptima
situacioncliente
tipocliente
tipoembarque
tipoevento
tipopago
tipousuario
usuario

3. Listado General De Tablas Con Columnas Del Modelo Entidad Relación.

Columnas de la Tabla "almacenentradadistribucion"
Name
Almacenentradadistribucioncve
relacionembarqueclave
clientedifesacve
tipoclientedifesacve
empresacve
tipoclientecve
situacionclientecve
puntoentregacve
rutacve
empleadocve
departamentocve
cargocve
estatusdistribucioncve
tipoembarquecve
almacenentradadistribucionnume
almacenentradadistribucioncant
amacenentradadistribucionimpor
almacenentradadistribucionobse
almacenentradadistribucionfoli
almacenentradadistribucionfech
almacenentradadistribucionfech
almacenentradadistribucionultf
almacenentradadistribucioneven

Columnas de la Tabla "almacenentradapublicacion"
Name
almacenentradacve
edicioncve
publicacioncve
proveedorcve
empresacve
compracve
empleadocve
departamentocve
cargocve
almacenentradafecha
almacenentradacantidad
almacenentradanumero

Columnas de la Tabla "bitacoraevento"
Name
bitacoraeventocve
rutacve
empleadocve
departamentocve
cargocve
tipoeventocve
bitacoraeventofecha
bitacoraeventodescripcion

Columnas de la Tabla "cargo"
Name
cargocve

Columnas de la Tabla "cargo"
Name
cargonombre
cargodesc

Columnas de la Tabla "clientedifesa"
Name
clientedifesacve
tipoclientedifesacve
empresacve
situacionclientecve
tipoclientecve
clientedifesanombre
clientedifesacontacto
clientedifesacontactocargo
clientedifesacpfisico
clientedifesanumerofisico
clientedifesacallefisico
clientedifesacoloniafisico
clientedifesaestadofisico
clientedifesacallefiscal
clientedifesanumerofiscal
clientedifesacoloniafiscal
clientedifesaestadofiscal
clientedifesacpfiscal
clientedifesarfc
clientedifesatelefono1
clientedifesatelefono2
clientedifesatelefono3
clientedifesafax
clientedifesaemail
clientedifesafechaalta
clientedifesafechabaja
clientedifesadiascredito

Columnas de la Tabla "clienteentrega"
Name
puntoentregacve
clientedifesacve
tipoclientedifesacve
empresacve
situacionclientecve
tipoclientecve
rutacve
empleadocve
departamentocve
cargocve

Columnas de la Tabla "compra"
Name
compracve
empleadocve
departamentocve
cargocve
proveedorcve
empresacve
comprafechaelaboracion
comprarecepcion
comprafechapago
comprareferencia
compradocumentopago
compraimporte

Columnas de la Tabla "comprapago"
Name
comprapagocve
tipopagocve
cuentabancocve
compracve
empleadocve
departamentocve
cargocve
proveedorcve
empresacve
comprapagofecha
comprapagoimporte
comprapagoreferencia

Columnas de la Tabla "cuentabanco"
Name
cuentabancocve
cuentabancosucursal
cuentabanconumero
cuentabanconombre
cuentabancotitular
cuentabancoCLABE
cuentabancoplaza
cuentabancoestado
cuentabancoobservaciones

Columnas de la Tabla "departamento"
Name
departamentocve
departamentonombre
departamentodesc

Columnas de la Tabla "descuento"
Name
descuentocve
clientedifesacve
tipoclientedifesacve
empresacve
situacionclientecve
tipoclientecve
publicacioncve
proveedorcve
descuentoconcepto
descuentodescripcion
descuentoporcentaje

Columnas de la Tabla "devolucion"
Name
devolucioncve
edicioncve
publicacioncve
proveedorcve
empresacve
clientedifesacve
tipoclientedifesacve
situacionclientecve
tipoclientecve
devolucionfoliodocumento
devolucionfechadocumento
devolucioncantidadcontada
devolucionacreditada
devolucioncantidadremision

Columnas de la Tabla "distribucioncosto"
Name
distribucioncostocve
clientedifesacve
tipoclientedifesacve
situacionclientecve
empresacve
tipoclientecve
tipoembarquecve
distribucioncostoimporte

Columnas de la Tabla "edicion"
Name
edicioncve
publicacioncve
proveedorcve
empresacve
edicionnombre
edicionfecha
ediciondescripcion
edicionfecharecepcion
edicionprecioportada
edicionsubtitulo
ediciondotacion
edicionpreciocompra

Columnas de la Tabla "empleado"
Name
empleadocve
departamentocve
cargocve
empleadonombre
empleadoapellidopat
empleadoapellidomat
empleadorfc
empleadoregimss
empleadolicenciaconducirnumero
empleadolicenciaconducirtipo

Columnas de la Tabla "empresa"
Name
empresacve
empresarazonsocial
empresacallefiscal
empresarfc
empresanumerofiscal
empresacoloniafiscal
empresamunipfiscal
empresaedofiscal
empresacpfiscal
empresacallefisico
empresanumerofisico
empresacoloniafisico
empresamunipfisico
empresaedofisico
empresacpfisico
empresareplegal
empresatelefono1
empresatelefono2
empresatelefono3
empresafax
empresamail

Columnas de la Tabla "empresa"
Name
empresafechaalta
empresafechabaja

Columnas de la Tabla "entrega"
Name
entregacve
facturapublicacionnumero
clientedifesacve
tipoclientedifesacve
empresacve
situacionclientecve
tipoclientecve
descuentocve
publicacioncve
proveedorcve
puntoentregacve
rutacve
empleadocve
departamentocve
cargocve
entregafecha
entregafolio

Columnas de la Tabla "estatusdistribucion"
Name
estatusdistribucioncve
estatusdistribucionnombre
estatusdistribuciondescripcion

Columnas de la Tabla "facturadistribucion"
Name
relacionembarqueclave
facturadistribucionnumero
clientedifesacve
tipoclientedifesacve
empresacve
situacionclientecve
tipoclientecve
tipoembarquecve
distribucioncostocve
facturadistribucioncantidad
facturadistribucionconcepto
facturadistribucionfechapago

Columnas de la Tabla "facturadistribucionpago"
Name
facturadistribucionpagocve
relacionembarqueclave
facturadistribucionnumero
clientedifesacve
tipoclientedifesacve
empresacve
situacionclientecve
tipoclientecve
tipoembarquecve
distribucioncostocve
tipopagocve
cuentabancocve
facturadistribucionpagoreferen
facturadistribucionpagofecha

Columnas de la Tabla "facturapublicacion"
Name
facturapublicacionnumero
clientedifesacve
tipoclientedifesacve
empresacve
situacionclientecve
tipoclientecve
descuentocve
publicacioncve
proveedorcve
puntoentregacve
rutacve
empleadocve
departamentocve
cargocve
facturapublicacionconcepto
facturapublicacionconceptocant
facturapublicacionconceptoimpo
facturapublicacionfechaemision
facturapublicacionfechapago

Columnas de la Tabla "facturapublicacionpago"
Name
facturapublicacionpagocve
tipopagocve
cuentabancocve
facturapublicacionpagoimporte
facturapublicacionpagofecha
facturapublicacionpagoreferenc

Columnas de la Tabla "phpSP_sessions"
Name
KEY_id
PRIMARY_KEY
UNIQUE_id_2
id
LastAction
ip
userID

Columnas de la Tabla "phpSP_sessionVars"
Name
UNIQUE_id
PRIMARY_KEY
id
session
name
intval
strval
KEYsessionID

Columnas de la Tabla "proveedor"
Name
proveedorcve
empresacve
proveedornombre
proveedorcontacto
proveedorcalle
proveedornumero
proveedorcolonia
proveedorestado
proveedorrfc

Columnas de la Tabla "proveedor"
Name
proveedortelefono1
proveedortelefono2
proveedorfax
proveedoremail
proveedorcumerocuenta
proveedorplazodevolucion
proveedorplazopago

Columnas de la Tabla "publicacion"
Name
publicacioncve
proveedorcve
empresacve
publicacionnombre
publicaciondescripcion
publicacionperiodo
publicacionpeso
publicacionIVA

Columnas de la Tabla "puntoentrega"
Name
puntoentregacve
rutacve
empleadocve
departamentocve
cargocve
puntoentregacp
puntoentreganombre
puntoentregaemail
puntoentregaestado
puntoentregafechaalta
puntoentregafax
puntoentregatelefono2
puntoentregatelefono1
puntoentregacolonia
puntoentreganumero
puntoentregacalle
puntoentregacontacto

Columnas de la Tabla "relacionembarque"
Name
relacionembarqueclave
clientedifesacve
tipoclientedifesacve
empresacve
situacionclientecve
tipoclientecve
tipoembarquecve
relacionembarquenumero
relacionembarquefecha
relacionembarqueobservacion

Columnas de la Tabla "ruta"
Name
rutacve
empleadocve
departamentocve
cargocve
rutanumero
rutadescripcion

Columnas de la Tabla "rutageneradaoptima"
Name
rutageneradaoptimacompid
rutacve
empleadocve
departamentocve
cargocve
puntoentregacve
rutageneradaoptimanumero
rutageneradaoptimaorden

Columnas de la Tabla "situacioncliente"
Name
situacionclientecve
situacionclientenombre
situacionclientedescripcion

Columnas de la Tabla "tipocliente"
Name
tipoclientecve
tipoclientenombre
tipoclientedescripcion

Columnas de la Tabla "tipoembarque"
Name
tipoembarquecve
tipoembarquenombre
tipoembarquedescripcion

Columnas de la Tabla "tipoevento"
Name
tipoeventocve
tipoeventonombre
tipoeventodescripcion
tipoeventonaturaleza

Columnas de la Tabla "tipopago"
Name
tipopagocve
tipopagonombre
tipopagodescripcion

Columnas de la Tabla "tipousuario"
Name
tipousuariocve
tipousuarionombre
tipousuariodescripcion

Columnas de la Tabla "usuario"
Name
usuariocveid
tipousuariocve
empleadocve
departamentocve
cargocve
usuarionombre
usuariopass

4. Esquema SQL de la Base de Datos.

```
CREATE TABLE almacenentradadistribucion (
    almacenentradadistribucioncve CHAR(18) NOT NULL,
    almacenentradadistribucionnume CHAR(18) NOT NULL,
    relacionembarqueclave CHAR(18) NOT NULL,
    clientedifesacve    INTEGER NOT NULL,
    tipoclientedifesacve INTEGER NOT NULL,
    empresacve          CHAR(18) NOT NULL,
    situacionclientecve  INTEGER NOT NULL,
    tipoclientecve      INTEGER NOT NULL,
    almacenentradadistribucioncant CHAR(18) NULL,
    amacenentradadistribucionimpor CHAR(18) NULL,
    almacenentradadistribucionobse CHAR(18) NULL,
    puntoentregacve     CHAR(18) NOT NULL,
    almacenentradadistribucionfoli CHAR(18) NULL,
    almacenentradadistribucionfech CHAR(18) NULL,
    rutacve             CHAR(18) NOT NULL,
    empleadocve         INTEGER NOT NULL,
    departamentocve     INTEGER NOT NULL,
    cargocve            INTEGER NOT NULL,
    estatusdistribucioncve CHAR(18) NOT NULL,
    almacenentradadistribucionfech CHAR(18) NULL,
    almacenentradadistribucionultf CHAR(18) NULL,
    tipoembarquecve     CHAR(18) NOT NULL,
    almacenentradadistribucioneven CHAR(18) NULL
);

ALTER TABLE almacenentradadistribucion
    ADD ( PRIMARY KEY (almacenentradadistribucioncve,
        relacionembarqueclave, clientedifesacve,
        tipoclientedifesacve, empresacve, tipoclientecve,
        situacionclientecve, puntoentregacve, rutacve,
        empleadocve, departamentocve, cargocve,
        estatusdistribucioncve, tipoembarquecve) ) ;

CREATE TABLE almacenentradapublicacion (
    almacenentradacve   CHAR(18) NOT NULL,
    almacenentradacantidad CHAR(18) NOT NULL,
    almacenentradafecha  CHAR(18) NOT NULL,
    edicioncve          CHAR(18) NOT NULL,
    publicacioncve      CHAR(18) NOT NULL,
    proveedorcve        CHAR(18) NOT NULL,
    empresacve          CHAR(18) NOT NULL,
    almacenentradanumero CHAR(18) NULL,
    compracve           CHAR(18) NOT NULL,
    empleadocve         INTEGER NOT NULL,
    departamentocve     INTEGER NOT NULL,
    cargocve            INTEGER NOT NULL
);

ALTER TABLE almacenentradapublicacion
    ADD ( PRIMARY KEY (almacenentradacve, edicioncve,
        publicacioncve, proveedorcve, empresacve, compracve,
        empleadocve, departamentocve, cargocve) ) ;

CREATE TABLE bitacoraevento (
    bitacoraeventocve   CHAR(18) NOT NULL,
    bitacoraeventofecha  CHAR(18) NULL,
    bitacoraeventodescripcion CHAR(18) NULL,
    rutacve             CHAR(18) NOT NULL,
```

```
    empleadocve         INTEGER NOT NULL,
    departamentocve     INTEGER NOT NULL,
    cargocve            INTEGER NOT NULL,
    tipoeventocve       CHAR(18) NOT NULL
);


ALTER TABLE bitacoraevento
    ADD ( PRIMARY KEY (bitacoraeventocve, rutacve, empleadocve,
        departamentocve, cargocve, tipoeventocve) ) ;


CREATE TABLE cargo (
    cargocve            INTEGER NOT NULL,
    cargonombre         CHAR(18) NULL,
    cargodesc           CHAR(18) NULL
);

ALTER TABLE cargo
    ADD ( PRIMARY KEY (cargocve) ) ;

CREATE TABLE clientedifesa (
    clientedifesacve    INTEGER NOT NULL,
    tipoclientedifesacve INTEGER NOT NULL,
    clientedifesanombre CHAR(18) NOT NULL,
    clientedifesacontacto CHAR(18) NOT NULL,
    clientedifcsacontactocargo CHAR(18) NOT NULL,
    clientedifesacpfisico CHAR(18) NOT NULL,
    clientedifesanumerofisico CHAR(18) NOT NULL,
    clientedifesacallcfisico CHAR(18) NOT NULL,
    clientedifesacoloniafisico CHAR(18) NOT NULL,
    clientedifesaestadofisico CHAR(18) NOT NULL,
    clientedifesacallefiscal CHAR(18) NOT NULL,
    clientedifesanumerofiscal CHAR(18) NOT NULL,
    clientedifesacoloniafiscal CHAR(18) NOT NULL,
    clientedifesaestadofiscal CHAR(18) NOT NULL,
    clientedifesacpfiscal CHAR(18) NOT NULL,
    clientedifesarfc    CHAR(18) NOT NULL,
    clientedifesatelefono1 CHAR(18) NOT NULL,
    clientedifesatelefono2 CHAR(18) NOT NULL,
    clientedifesatelefono3 CHAR(18) NOT NULL,
    clientedifesafax    CHAR(18) NOT NULL,
    clientedifesaemail  CHAR(18) NOT NULL,
    clientedifesafechaalta CHAR(18) NOT NULL,
    clientedifesafechabaja CHAR(18) NOT NULL,
    clientedifesadiascredito CHAR(18) NOT NULL,
    empresacve          CHAR(18) NOT NULL,
    situacionclientecve INTEGER NOT NULL,
    tipoclientecve      INTEGER NOT NULL
);

ALTER TABLE clientedifesa
    ADD ( PRIMARY KEY (clientedifesacve, tipoclientedifesacve,
        empresacve, situacionclientecve, tipoclientecve) ) ;

CREATE TABLE clienteentrega (
    puntoentregacve     CHAR(18) NOT NULL,
    clientedifesacve    INTEGER NOT NULL,
    tipoclientedifesacve INTEGER NOT NULL,
    empresacve          CHAR(18) NOT NULL,
    situacionclientecve INTEGER NOT NULL,
    tipoclientecve      INTEGER NOT NULL,
```

```
    rutacve          CHAR(18) NOT NULL,
    empleadocve      INTEGER NOT NULL,
    departamentocve  INTEGER NOT NULL,
    cargocve         INTEGER NOT NULL
);

ALTER TABLE clienteentrega
    ADD ( PRIMARY KEY (puntoentregacve, clientedifesacve,
        tipoclientedifesacve, empresacve, situacionclientecve,
        tipoclientecve, rutacve, empleadocve, departamentocve,
        cargocve) ) ;


CREATE TABLE compra (
    compracve          CHAR(18) NOT NULL,
    empleadocve        INTEGER NOT NULL,
    departamentocve    INTEGER NOT NULL,
    cargocve           INTEGER NOT NULL,
    comprafechaelaboracion CHAR(18) NULL,
    proveedorcve       CHAR(18) NOT NULL,
    empresacve         CHAR(18) NOT NULL,
    comprarecepcion    CHAR(18) NULL,
    comprafechapago    CHAR(18) NULL,
    compradocumentopago CHAR(18) NULL,
    compraimporte      CHAR(18) NULL,
    comprareferencia   CHAR(18) NULL
);

ALTER TABLE compra
    ADD ( PRIMARY KEY (compracve, empleadocve, departamentocve,
        cargocve, proveedorcve, empresacve) ) ;

CREATE TABLE comprapago (
    comprapagocve      CHAR(18) NOT NULL,
    comprapagofecha    CHAR(18) NOT NULL,
    comprapagoimporte  CHAR(18) NOT NULL,
    tipopagocve        CHAR(18) NOT NULL,
    cuentabancocve     CHAR(18) NOT NULL,
    comprapagoreferencia CHAR(18) NULL,
    compracve          CHAR(18) NOT NULL,
    empleadocve        INTEGER NOT NULL,
    departamentocve    INTEGER NOT NULL,
    cargocve           INTEGER NOT NULL,
    proveedorcve       CHAR(18) NOT NULL,
    empresacve         CHAR(18) NOT NULL
);

ALTER TABLE comprapago
    ADD ( PRIMARY KEY (comprapagocve, tipopagocve, cuentabancocve,
        compracve, empleadocve, departamentocve, cargocve,
        proveedorcve, empresacve) ) ;

CREATE TABLE cuentabanco (
    cuentabancocve     CHAR(18) NOT NULL,
    cuentabanconumero  CHAR(18) NOT NULL,
    cuentabanconombre  CHAR(18) NOT NULL,
    cuentabancotitular CHAR(18) NOT NULL,
    cuentabancosucursal CHAR(18) NOT NULL,
    cuentabancoplaza   CHAR(18) NOT NULL,
    cuentabancoestado  CHAR(18) NOT NULL,
    cuentabancoCLABE   CHAR(18) NOT NULL,
    cuentabancoobservaciones CHAR(18) NOT NULL
```

```
);

ALTER TABLE cuentabanco
    ADD ( PRIMARY KEY (cuentabancocve) ) ;


CREATE TABLE departamento (
    departamentocve     INTEGER NOT NULL,
    departamentonombre  CHAR(18) NOT NULL,
    departamentodesc    CHAR(18) NOT NULL
);


ALTER TABLE departamento
    ADD ( PRIMARY KEY (departamentocve) ) ;


CREATE TABLE descuento (
    descuentocve        CHAR(18) NOT NULL,
    clientedifesacve    INTEGER NOT NULL,
    tipoclientedifesacve INTEGER NOT NULL,
    empresacve          CHAR(18) NOT NULL,
    situacionclientecve INTEGER NOT NULL,
    tipoclientecve      INTEGER NOT NULL,
    publicacioncve      CHAR(18) NOT NULL,
    proveedorcve        CHAR(18) NOT NULL,
    descuentoconcepto   CHAR(18) NULL,
    descuentodescripcion CHAR(18) NULL,
    descuentoporcentaje CHAR(18) NULL
);


ALTER TABLE descuento
    ADD ( PRIMARY KEY (descuentocve, clientedifesacve,
        tipoclientedifesacve, empresacve, situacionclientecve,
        tipoclientecve, publicacioncve, proveedorcve) ) ;


CREATE TABLE devolucion (
    devolucioncve       CHAR(18) NOT NULL,
    edicioncve          CHAR(18) NOT NULL,
    publicacioncve      CHAR(18) NOT NULL,
    proveedorcve        CHAR(18) NOT NULL,
    empresacve          CHAR(18) NOT NULL,
    clientedifesacve    INTEGER NOT NULL,
    tipoclientedifesacve INTEGER NOT NULL,
    situacionclientecve INTEGER NOT NULL,
    tipoclientecve      INTEGER NOT NULL,
    devolucionfoliodocumento CHAR(18) NOT NULL,
    devolucionfechadocumento CHAR(18) NOT NULL,
    devolucioncantidadcontada CHAR(18) NULL,
    devolucionacreditada CHAR(18) NULL,
    devolucioncantidadremision CHAR(18) NULL
);


ALTER TABLE devolucion
    ADD ( PRIMARY KEY (devolucioncve, edicioncve, publicacioncve,
        proveedorcve, empresacve, clientedifesacve,
        tipoclientedifesacve, situacionclientecve,
        tipoclientecve) ) ;
```

```
CREATE TABLE distribucioncosto (
    distribucioncostocve CHAR(18) NOT NULL,
    clientedifesacve    INTEGER NOT NULL,
    tipoclientedifesacve INTEGER NOT NULL,
    empresacve          CHAR(18) NOT NULL,
    situacionclientecve  INTEGER NOT NULL,
    tipoclientecve      INTEGER NOT NULL,
    tipoembarquecve     CHAR(18) NOT NULL,
    distribucioncostoimporte CHAR(18) NULL
);


ALTER TABLE distribucioncosto
    ADD ( PRIMARY KEY (distribucioncostocve, clientedifesacve,
        tipoclientedifesacve, situacionclientecve, empresacve,
        tipoclientecve, tipoembarquecve) ) ;


CREATE TABLE edicion (
    edicioncve          CHAR(18) NOT NULL,
    edicionnombre       CHAR(18) NULL,
    edicionfecha        CHAR(18) NULL,
    ediciondescripcion  CHAR(18) NULL,
    edicionfecharecepcion CHAR(18) NULL,
    edicionprecioportada CHAR(18) NULL,
    edicionsubtitulo    CHAR(18) NULL,
    publicacioncve      CHAR(18) NOT NULL,
    proveedorcve        CHAR(18) NOT NULL,
    empresacve          CHAR(18) NOT NULL,
    ediciondotacion     CHAR(18) NULL,
    edicionpreciocompra  CHAR(18) NULL
);


ALTER TABLE edicion
    ADD ( PRIMARY KEY (edicioncve, publicacioncve, proveedorcve,
        empresacve) ) ;


CREATE TABLE empleado (
    empleadocve         INTEGER NOT NULL,
    empleadonombre      CHAR(18) NOT NULL,
    empleadoapellidomat  CHAR(18) NOT NULL,
    empleadoapellidopat  CHAR(18) NOT NULL,
    empleadorfc         CHAR(18) NOT NULL,
    empleadoregimss     CHAR(18) NOT NULL,
    departamentocve     INTEGER NOT NULL,
    cargocve            INTEGER NOT NULL,
    empleadolicenciaconducirnumero CHAR(18) NULL,
    empleadolicenciaconducirtipo CHAR(18) NULL
);


ALTER TABLE empleado
    ADD ( PRIMARY KEY (empleadocve, departamentocve, cargocve) ) ;


CREATE TABLE empresa (
    empresacve          CHAR(18) NOT NULL,
    empresarazonsocial   CHAR(18) NOT NULL,
```

```
    empresarfc          CHAR(18) NOT NULL,
    empresacallefiscal  CHAR(18) NULL,
    empresanumerofiscal CHAR(18) NULL,
    empresacoloniafiscal CHAR(18) NULL,
    empresamunipfiscal  CHAR(18) NULL,
    empresacpfiscal     CHAR(18) NULL,
    empresacallefisico  CHAR(18) NULL,
    empresanumerofisico CHAR(18) NULL,
    empresacoloniafisico CHAR(18) NULL,
    empresamunipfisico  CHAR(18) NULL,
    empresaedofisico    CHAR(18) NULL,
    empresareplegal     CHAR(18) NULL,
    empresaedofiscal    CHAR(18) NULL,
    emppresacpfisico    CHAR(18) NULL,
    empresatelefono1    CHAR(18) NULL,
    empresatelefono2    CHAR(18) NULL,
    empresatelefono3    CHAR(18) NULL,
    empresafax          CHAR(18) NULL,
    empresamail         CHAR(18) NULL,
    empresafechaalta    CHAR(18) NULL,
    empresafechabaja    CHAR(18) NULL
);

ALTER TABLE empresa
    ADD ( PRIMARY KEY (empresacve) ) ;

CREATE TABLE entrega (
    entregacve          CHAR(18) NOT NULL,
    entregafecha        CHAR(18) NULL,
    entregafolio        CHAR(18) NULL,
    facturapublicacionnumero CHAR(18) NOT NULL,
    clientedifesacve    INTEGER NOT NULL,
    tipoclientedifesacve INTEGER NOT NULL,
    empresacve          CHAR(18) NOT NULL,
    situacionclientecve INTEGER NOT NULL,
    tipoclientecve      INTEGER NOT NULL,
    descuentocve        CHAR(18) NOT NULL,
    publicacioncve      CHAR(18) NOT NULL,
    proveedorcve        CHAR(18) NOT NULL,
    puntoentregacve     CHAR(18) NOT NULL,
    rutacve             CHAR(18) NOT NULL,
    empleadocve         INTEGER NOT NULL,
    departamentocve     INTEGER NOT NULL,
    cargocve            INTEGER NOT NULL
);

ALTER TABLE entrega
    ADD ( PRIMARY KEY (entregacve, facturapublicacionnumero,
        clientedifesacve, tipoclientedifesacve, empresacve,
        situacionclientecve, tipoclientecve, descuentocve,
        publicacioncve, proveedorcve, puntoentregacve, rutacve,
        empleadocve, departamentocve, cargocve) ) ;

CREATE TABLE estatusdistribucion (
    estatusdistribucioncve CHAR(18) NOT NULL,
    estatusdistribucionnombre CHAR(18) NOT NULL,
    estatusdistribuciondescripcion CHAR(18) NOT NULL
);
```

```
ALTER TABLE estatusdistribucion
    ADD ( PRIMARY KEY (estatusdistribucioncve) ) ;


CREATE TABLE facturadistribucion (
    facturadistribucionnumero CHAR(18) NOT NULL,
    facturadistribucioncantidad CHAR(18) NULL,
    relacionembarqueclave CHAR(18) NOT NULL,
    clientedifesacve    INTEGER NOT NULL,
    tipoclientedifesacve INTEGER NOT NULL,
    empresacve          CHAR(18) NOT NULL,
    situacionclientecve INTEGER NOT NULL,
    tipoclientecve      INTEGER NOT NULL,
    tipoembarquecve     CHAR(18) NOT NULL,
    distribucioncostocve CHAR(18) NOT NULL,
    facturadistribucionconcepto CHAR(18) NULL,
    facturadistribucionfecbapago CHAR(18) NULL
);


ALTER TABLE facturadistribucion
    ADD ( PRIMARY KEY (relacionembarqueclave,
        facturadistribucionnumero, clientedifesacve,
        tipoclientedifesacve, empresacve, situacionclientecve,
        tipoclientecve, tipoembarquecve, distribucioncostocve) ) ;


CREATE TABLE facturadistribucionpago (
    facturadistribucionpagocve CHAR(18) NOT NULL,
    facturadistribucionpagofecha CHAR(18) NOT NULL,
    facturadistribucionpagoreferen CHAR(18) NOT NULL,
    relacionembarqueclave CHAR(18) NOT NULL,
    facturadistribucionnumero CHAR(18) NOT NULL,
    clientedifesacve    INTEGER NOT NULL,
    tipoclientedifesacve INTEGER NOT NULL,
    empresacve          CHAR(18) NOT NULL,
    situacionclientecve INTEGER NOT NULL,
    tipoclientecve      INTEGER NOT NULL,
    tipoembarquecve     CHAR(18) NOT NULL,
    distribucioncostocve CHAR(18) NOT NULL,
    tipopagocve         CHAR(18) NOT NULL,
    cuentabancocve      CHAR(18) NOT NULL
);


ALTER TABLE facturadistribucionpago
    ADD ( PRIMARY KEY (facturadistribucionpagocve,
        relacionembarqueclave, facturadistribucionnumero,
        clientedifesacve, tipoclientedifesacve, empresacve,
        situacionclientecve, tipoclientecve, tipoembarquecve,
        distribucioncostocve, tipopagocve, cuentabancocve) ) ;


CREATE TABLE facturapublicacion (
    facturapublicacionnumero CHAR(18) NOT NULL,
    facturapublicacionconcepto CHAR(18) NULL,
    clientedifesacve    INTEGER NOT NULL,
    tipoclientedifesaeve INTEGER NOT NULL,
    empresacve          CHAR(18) NOT NULL,
    situacionclientecve INTEGER NOT NULL,
```

```
    tipoclientecve      INTEGER NOT NULL,
    facturapublicacionconceptocant CHAR(18) NULL,
    facturapublicacionconceptoimpo CHAR(18) NULL,
    descuentocve        CHAR(18) NOT NULL,
    facturapublicacionfechaemision CHAR(18) NULL,
    facturapublicacionfechapago CHAR(18) NULL,
    puntoentregacve     CHAR(18) NOT NULL,
    publicacioncve      CHAR(18) NOT NULL,
    proveedorcve        CHAR(18) NOT NULL,
    rutacve             CHAR(18) NOT NULL,
    empleadocve         INTEGER NOT NULL,
    departamentocve     INTEGER NOT NULL,
    cargocve            INTEGER NOT NULL
);


ALTER TABLE facturapublicacion
    ADD ( PRIMARY KEY (facturapublicacionnumero, clientedifesacve,
        tipoclientedifesacve, empresacve, situacionclientecve,
        tipoclientecve, descuentocve, publicacioncve,
        proveedorcve, puntoentregacve, rutacve, empleadocve,
        departamentocve, cargocve) ) ;


CREATE TABLE facturapublicacionpago (
    facturapublicacionpagocve CHAR(18) NOT NULL,
    facturapublicacionpagofecha CHAR(18) NOT NULL,
    facturapublicacionpagoimporte CHAR(18) NOT NULL,
    facturapublicacionpagoreferenc CHAR(18) NOT NULL,
    tipopagocve         CHAR(18) NOT NULL,
    cuentabancocve      CHAR(18) NOT NULL
);


ALTER TABLE facturapublicacionpago
    ADD ( PRIMARY KEY (facturapublicacionpagocve, tipopagocve,
        cuentabancocve) ) ;


CREATE TABLE phpSP_sessions (
    id              CHAR(18) NOT NULL,
    LastAction      CHAR(18) NULL,
    ip              CHAR(18) NULL,
    userID          CHAR(18) NULL,
    PRIMARY_KEY     CHAR(18) NOT NULL,
    KEY_id          CHAR(18) NOT NULL,
    UNIQUE_id_2     CHAR(18) NOT NULL
);


ALTER TABLE phpSP_sessions
    ADD ( PRIMARY KEY (KEY_id, PRIMARY_KEY, UNIQUE_id_2) ) ;


CREATE TABLE phpSP_sessionVars (
    id              CHAR(18) NOT NULL,
    session         CHAR(18) NOT NULL,
    name            CHAR(18) NULL,
    intval          CHAR(18) NULL,
    strval          CHAR(18) NULL,
    PRIMARY_KEY     CHAR(18) NOT NULL,
    KEYsessionID    CHAR(18) NULL,
```

```
    UNIQUE_id         CHAR(18) NOT NULL
);


ALTER TABLE phpSP_sessionVars
    ADD ( PRIMARY KEY (UNIQUE_id, PRIMARY_KEY) ) ;


CREATE TABLE proveedor (
    proveedorcve       CHAR(18) NOT NULL,
    proveedornombre    CHAR(18) NULL,
    proveedorcontacto  CHAR(18) NULL,
    proveedorcalle     CHAR(18) NULL,
    proveedornumero    CHAR(18) NULL,
    proveedorcolonia   CHAR(18) NULL,
    proveedorestado    CHAR(18) NULL,
    proveedorrfc       CHAR(18) NULL,
    proveedortelefono1 CHAR(18) NULL,
    proveedortelefono2 CHAR(18) NULL,
    proveedorfax       CHAR(18) NULL,
    proveedoremail     CHAR(18) NULL,
    empresacve         CHAR(18) NOT NULL,
    proveedorcumerocuenta CHAR(18) NULL,
    proveedorplazodevolucion CHAR(18) NULL,
    proveedorplazopago CHAR(18) NULL
);


ALTER TABLE proveedor
    ADD ( PRIMARY KEY (proveedorcve, empresacve) ) ;


CREATE TABLE publicacion (
    publicacioncve     CHAR(18) NOT NULL,
    publicacionnombre  CHAR(18) NULL,
    publicaciondescripcion CHAR(18) NULL,
    proveedorcve       CHAR(18) NOT NULL,
    empresacve         CHAR(18) NOT NULL,
    publicacionperiodo CHAR(18) NULL,
    publicacionpeso    CHAR(18) NULL,
    publicacionIVA     CHAR(18) NULL
);


ALTER TABLE publicacion
    ADD ( PRIMARY KEY (publicacioncve, proveedorcve, empresacve) ) ;


CREATE TABLE puntoentrega (
    puntoentreganombre CHAR(18) NOT NULL,
    puntoentregacontacto CHAR(18) NOT NULL,
    puntoentregacp     CHAR(18) NOT NULL,
    puntoentreganumero CHAR(18) NOT NULL,
    puntoentregacve    CHAR(18) NOT NULL,
    puntoentregacalle  CHAR(18) NOT NULL,
    puntoentregacolonia CHAR(18) NOT NULL,
    puntoentregaestado CHAR(18) NOT NULL,
    puntoentregatelefono1 CHAR(18) NOT NULL,
    puntoentregatelefono2 CHAR(18) NOT NULL,
    puntoentregafax    CHAR(18) NOT NULL,
    puntoentregaemail  CHAR(18) NOT NULL,
    puntoentregafechaalta CHAR(18) NOT NULL,
```

```
    rutacve          CHAR(18) NOT NULL,
    empleadocve      INTEGER NOT NULL,
    departamentocve  INTEGER NOT NULL,
    cargocve         INTEGER NOT NULL
);


ALTER TABLE puntoentrega
    ADD ( PRIMARY KEY (puntoentregacve, rutacve, empleadocve,
        departamentocve, cargocve) ) ;


CREATE TABLE relacionembarque (
    relacionembarqueclave CHAR(18) NOT NULL,
    relacionembarquenumero CHAR(18) NOT NULL,
    relacionembarquefecha CHAR(18) NOT NULL,
    clientedifesacve   INTEGER NOT NULL,
    tipoclientedifesacve INTEGER NOT NULL,
    empresacve         CHAR(18) NOT NULL,
    situacionclientecve INTEGER NOT NULL,
    tipoclientecve     INTEGER NOT NULL,
    relacionembarqueobservacion CHAR(18) NULL,
    tipoembarquecve    CHAR(18) NOT NULL
);


ALTER TABLE relacionembarque
    ADD ( PRIMARY KEY (relacionembarqueclave, clientedifesacve,
        tipoclientedifesacve, empresacve, situacionclientecve,
        tipoclientecve, tipocmbarquecve) ) ;


CREATE TABLE ruta (
    rutacve          CHAR(18) NOT NULL,
    rutanumero       CHAR(18) NOT NULL,
    rutadescripcion  CHAR(18) NOT NULL,
    empleadocve      INTEGER NOT NULL,
    departamentocve  INTEGER NOT NULL,
    cargocve         INTEGER NOT NULL
);


ALTER TABLE ruta
    ADD ( PRIMARY KEY (rutacve, empleadocve, departamentocve,
        eargocve) ) ;


CREATE TABLE rutageneradaoptima (
    rutageneradaoptimacompid CHAR(18) NOT NULL,
    rutageneradaoptimanumero CHAR(18) NULL,
    rutageneradaoptimaorden CHAR(18) NULL,
    rutacve          CHAR(18) NOT NULL,
    empleadocve      INTEGER NOT NULL,
    departamentocve  INTEGER NOT NULL,
    cargocve         INTEGER NOT NULL,
    puntoentregacve  CHAR(18) NOT NULL
);


ALTER TABLE rutageneradaoptima
    ADD ( PRIMARY KEY (rutageneradaoptimacompid, rutacve,
        empleadocve, departamentocve, cargocve, puntoentregacve) ) ;
```

```
CREATE TABLE situacioncliente (
    situacionclientecve INTEGER NOT NULL,
    situacionclientenombre CHAR(18) NOT NULL,
    situacionclientedescripcion CHAR(18) NOT NULL
);

ALTER TABLE situacioncliente
    ADD ( PRIMARY KEY (situacionclientecve) ) ;

CREATE TABLE tipocliente (
    tipoclientecve      INTEGER NOT NULL,
    tipoclientenombre   CHAR(18) NOT NULL,
    tipoclientedescripcion CHAR(18) NOT NULL
);

ALTER TABLE tipocliente
    ADD ( PRIMARY KEY (tipoclientecve) ) ;

CREATE TABLE tipoembarque (
    tipoembarquecve     CHAR(18) NOT NULL,
    tipoembarquenombre  CHAR(18) NOT NULL,
    tipoembarquedescripcion CHAR(18) NOT NULL
);

ALTER TABLE tipoembarque
    ADD ( PRIMARY KEY (tipoembarquecve) ) ;

CREATE TABLE tipoevento (
    tipoeventocve       CHAR(18) NOT NULL,
    tipoeventonombre    CHAR(18) NULL,
    tipoeventodescripcion CHAR(18) NULL,
    tipoeventonaturaleza CHAR(18) NULL
);

ALTER TABLE tipoevento
    ADD ( PRIMARY KEY (tipoeventocve) ) ;

CREATE TABLE tipopago (
    tipopagocve         CHAR(18) NOT NULL,
    tipopagonombre      CHAR(18) NULL,
    tipopagodescripcion  CHAR(18) NULL
);

ALTER TABLE tipopago
    ADD ( PRIMARY KEY (tipopagocve) ) ;

CREATE TABLE tipousuario (
    tipousuariocve      CHAR(18) NOT NULL,
    tipousuarionombre   CHAR(18) NULL,
    tipousuariodescripcion CHAR(18) NULL
```

```
);


ALTER TABLE tipousuario
    ADD ( PRIMARY KEY (tipousuariocve) ) ;


CREATE TABLE usuario (
    usuariocveid        CHAR(18) NOT NULL,
    usuarionombre       CHAR(18) NULL,
    usuariopass         CHAR(18) NULL,
    tipousuariocve      CHAR(18) NOT NULL,
    empleadocve         INTEGER NOT NULL,
    departamentocve     INTEGER NOT NULL,
    cargocve            INTEGER NOT NULL
),


ALTER TABLE usuario
    ADD ( PRIMARY KEY (usuariocveid, tipousuariocve, empleadocve,
        departamentocve, cargocve) ) ,


ALTER TABLE almacenentradadistribucion
    ADD ( FOREIGN KEY (estatusdistribucioncve)
                REFERENCES estatusdistribucion ) ;


ALTER TABLE almacenentradadistribucion
    ADD ( FOREIGN KEY (puntoentregacve, clientedifesacve,
        tipoclientedifesacve, empresacve, situacionclientecve,
        tipoclientecve, rutacve, empleadocve, departamentocve,
        cargocve)
                REFERENCES clienteentrega ) ;


ALTER TABLE almacenentradadistribucion
    ADD ( FOREIGN KEY (relacionembarqueclave, clientedifesacve,
        tipoclientedifesacve, empresacve, situacionclientecve,
        tipoclientecve, tipoembarquecve)
                REFERENCES relacionembarque ) ;


ALTER TABLE almacenentradapublicacion
    ADD ( FOREIGN KEY (compracve, empleadocve, departamentocve,
        cargocve, proveedorcve, empresacve)
                REFERENCES compra ) ;


ALTER TABLE almacenentradapublicacion
    ADD ( FOREIGN KEY (edicioncve, publicacioncve, proveedorcve,
        empresacve)
                REFERENCES edicion ) ,


ALTER TABLE bitacoraevento
    ADD ( FOREIGN KEY (tipoeventocve)
                REFERENCES tipoevento ) ;


ALTER TABLE bitacoraevento
    ADD ( FOREIGN KEY (rutacve, empleadocve, departamentocve,
```

```
    cargocve)
            REFERENCES ruta ) ;

ALTER TABLE clientedifesa
    ADD ( FOREIGN KEY (tipoclientecve)
            REFERENCES tipocliente ) ;

ALTER TABLE clientedifesa
    ADD ( FOREIGN KEY (situacionclientecve)
            REFERENCES situacioncliente ) ;

ALTER TABLE clientedifesa
    ADD ( FOREIGN KEY (empresacve)
            REFERENCES empresa ) ;

ALTER TABLE clienteentrega
    ADD ( FOREIGN KEY (clientedifesacve, tipoclientedifesacve,
        empresacve, situacionclientecve, tipoclientecve)
            REFERENCES clientedifesa ) ;

ALTER TABLE clienteentrega
    ADD ( FOREIGN KEY (puntoentregacve, rutacve, empleadocve,
        departamentocve, cargocve)
            REFERENCES puntoentrega ) ;

ALTER TABLE compra
    ADD ( FOREIGN KEY (proveedorcve, empresacve)
            REFERENCES proveedor ) ;

ALTER TABLE compra
    ADD ( FOREIGN KEY (empleadocve, departamentocve, cargocve)
            REFERENCES empleado ) ;

ALTER TABLE comprapago
    ADD ( FOREIGN KEY (compracve, empleadocve, departamentocve,
        cargocve, proveedorcve, empresacve)
            REFERENCES compra ) ;

ALTER TABLE comprapago
    ADD ( FOREIGN KEY (cuentabancocve)
            REFERENCES cuentabanco ) ;

ALTER TABLE comprapago
    ADD ( FOREIGN KEY (tipopagocve)
            REFERENCES tipopago ) ;

ALTER TABLE descuento
    ADD ( FOREIGN KEY (publicacioncve, proveedorcve, empresacve)
            REFERENCES publicacion ) ;
```

```
ALTER TABLE descuento
    ADD ( FOREIGN KEY (clientedifesacve, tipoclientedifesacve,
        empresacve, situacionclientecve, tipoclientecve)
                REFERENCES clientedifesa ) ;


ALTER TABLE devolucion
    ADD ( FOREIGN KEY (clientedifesacve, tipoclientedifesacve,
        empresacve, situacionclientecve, tipoclientecve)
                REFERENCES clientedifesa ) ;


ALTER TABLE devolucion
    ADD ( FOREIGN KEY (edicioncve, publicacioncve, proveedorcve,
        empresacve)
                REFERENCES edicion ) ;


ALTER TABLE distribucioncosto
    ADD ( FOREIGN KEY (tipoembarquecve)
                REFERENCES tipoembarque ) ;


ALTER TABLE distribucioncosto
    ADD ( FOREIGN KEY (clientedifesacve, tipoclientedifesacve,
        empresacve, situacionclientecve, tipoclientecve)
                REFERENCES clientedifesa ) ;


ALTER TABLE edicion
    ADD ( FOREIGN KEY (publicacioncve, proveedorcve, empresacve)
                REFERENCES publicacion ) ;


ALTER TABLE empleado
    ADD ( FOREIGN KEY (cargocve)
                REFERENCES cargo ) ;


ALTER TABLE empleado
    ADD ( FOREIGN KEY (departamentocve)
                REFERENCES departamento ) ;


ALTER TABLE entrega
    ADD ( FOREIGN KEY (facturapublicacionnumero, clientedifesacve,
        tipoclientedifesacve, empresacve, situacionclientecve,
        tipoclientecve, descuentocve, publicacioncve,
        proveedorcve, puntoentregacve, rutacve, empleadocve,
        departamentocve, cargocve)
                REFERENCES facturapublicacion ) ;


ALTER TABLE facturadistribucion
    ADD ( FOREIGN KEY (distribucioncostocve, clientedifesacve,
        tipoclientedifesacve, situacionclientecve, empresacve,
        tipoclientecve, tipoembarquecve)
                REFERENCES distribucioncosto ) ;


ALTER TABLE facturadistribucion
    ADD ( FOREIGN KEY (relacionembarqueclave, clientedifesacve,
```

```
        tipoclientedifesacve, empresacve, situacionclientecve,
        tipoclientecve, tipoembarquecve)
                REFERENCES relacionembarque ) ;


ALTER TABLE facturadistribucionpago
    ADD ( FOREIGN KEY (cuentabancocve)
                REFERENCES cuentabanco ) ;


ALTER TABLE facturadistribucionpago
    ADD ( FOREIGN KEY (tipopagocve)
                REFERENCES tipopago ) ;


ALTER TABLE facturadistribucionpago
    ADD ( FOREIGN KEY (relacionembarqueclave,
        facturadistribucionnumero, clientedifesacve,
        tipoclientedifesacve, empresacve, situacionclientecve,
        tipoclientecve, tipoembarquecve, distribucioncostocve)
                REFERENCES facturadistribucion ) ;


ALTER TABLE facturapublicacion
    ADD ( FOREIGN KEY (puntoentregacve, rutacve, empleadocve,
        departamentocve, cargocve)
                REFERENCES puntoentrega ) ;


ALTER TABLE facturapublicacion
    ADD ( FOREIGN KEY (descuentocve, clientedifesacve,
        tipoclientedifesacve, empresacve, situacionclientecve,
        tipoclientecve, publicacioncve, proveedorcve)
                REFERENCES descuento ) ;


ALTER TABLE facturapublicacion
    ADD ( FOREIGN KEY (clientedifesacve, tipoclientedifesacve,
        empresacve, situacionclientecve, tipoclientecve)
                REFERENCES clientedifesa ) ;


ALTER TABLE facturapublicacionpago
    ADD ( FOREIGN KEY (cuentabancocve)
                REFERENCES cuentabanco ) ;


ALTER TABLE facturapublicacionpago
    ADD ( FOREIGN KEY (tipopagocve)
                REFERENCES tipopago ) ;


ALTER TABLE proveedor
    ADD ( FOREIGN KEY (empresacve)
                REFERENCES empresa ) ;


ALTER TABLE publicacion
    ADD ( FOREIGN KEY (proveedorcve, empresacve)
                REFERENCES proveedor ) ;
```

```
ALTER TABLE puntoentrega
    ADD ( FOREIGN KEY (rutacve, empleadocve, departamentocve,
        cargocve)
                REFERENCES ruta ) ;


ALTER TABLE relacionembarque
    ADD ( FOREIGN KEY (tipoembarquecve)
                REFERENCES tipoembarque ) ;


ALTER TABLE relacionembarque
    ADD ( FOREIGN KEY (clientedifesacve, tipoclientedifesacve,
        empresacve, situacionclientecve, tipoclientecve)
                REFERENCES clientedifesa ) ;


ALTER TABLE ruta
    ADD ( FOREIGN KEY (empleadocve, departamentocve, cargocve)
                REFERENCES empleado ) ;


ALTER TABLE rutageneradaoptima
    ADD ( FOREIGN KEY (puntoentregacve, rutacve, empleadocve,
        departamentocve, cargocve)
                REFERENCES puntoentrega ) ;


ALTER TABLE rutageneradaoptima
    ADD ( FOREIGN KEY (rutacve, empleadocve, departamentocve,
        cargocve)
                REFERENCES ruta ) ;


ALTER TABLE usuario
    ADD ( FOREIGN KEY (empleadocve, departamentocve, cargocve)
                REFERENCES empleado ) ;


ALTER TABLE usuario
    ADD ( FOREIGN KEY (tipousuariocve)
                REFERENCES tipousuario ) ;

create trigger tI_almacenentradadistribucion after INSERT on almacenentradadistribucion for each row

-- INSERT trigger on almacenentradadistribucion
declare numrows INTEGER;
begin

  /* estatusdistribucion R/49 almacenentradadistribucion ON CHILD INSERT RESTRICT */
  select count(*) into numrows
    from estatusdistribucion
    where
      /* :new.estatusdistribucioncve = estatusdistribucion.estatusdistribucioncve */
      :new.estatusdistribucioncve = estatusdistribucion.estatusdistribucioncve;
  if (
    /* */

    numrows = 0
  )
  then
    raise_application_error(
```

```
  -20002,
  'Cannot INSERT almacenentradadistribucion because estatusdistribucion does not exist.'
 );
end if;


/* clienteentrega R/48 almacencntradadistribucion ON CHILD INSERT RESTRICT */
select count(*) into numrows
 from clienteentrega
 where
  /* :new.puntoentregacve = clienteentrega.puntoentregacve and
     :new.clicntedifesacve = clienteentrega.clientedifesacve and
     :new.tipoclientedifesacve = clienteentrega.tipoclientedifesacve and
     :new.empresacve = clienteentrega.empresacve and
     :new.situacionclientecve = clienteentrega.situacionclientecve and
     :new.tipoclientecve = clienteentrega.tipoclientecve and
     :new.rutacve = clienteentrega.rutacve and
     :new.empleadocve = clienteentrega.empleadocve and
     :new.departamentocve = clienteentrega.departamentocve and
     :new.cargocve = clienteentrega.cargocve */
   :new.puntoentregacve = clienteentrega.puntoentregacve and
   :new.clientedifesacve = clienteentrega.clientedifesacve and
   :new.tipoclientedifesacve = clientcentrega.tipoclientedifesacve and
   :new.empresacve = clienteentrega.empresacve and
   :new.situacionclientecve = clienteentrega.situacionclientecve and
   :new.tipoclientecve = clienteentrega.tipoclientecve and
   :new.rutacve = clienteentrega.rutacve and
   :new.empleadocve = clientcentrega.empleadocve and
   :new.departamentocve = clienteentrega.departamentocve and
   :new.cargocve = clienteentrega.cargocve;
if (
 /* */

 numrows = 0
)
then
 raise_application_error(
  -20002,
  'Cannot INSERT almacenentradadistribucion because clienteentrega does not exist.'
 );
end if;


/* relacionembarque R/46 almacenentradadistribucion ON CHILD INSERT RESTRICT */
select count(*) into numrows
 from relacionembarque
 where
  /* :new.relacionembarqueclave = relacionembarque.relacionembarqueclave and
     :new.clientedifesacve = relacionembarque.clientedifesacve and
     :new.tipoclientedifesacve = relacionembarque.tipoclientedifesacve and
     :new.empresacve = relacionembarque.empresacve and
     :new.situacionclientecve = relacionembarque.situacionclientecve and
     :new.tipoclientecve = relacionembarque.tipoclientecve and
     :new.tipoembarquecve = relacionembarque.tipoembarquecve */
   :new.relacionembarqueclave = relacioncmbarque.relacionembarqueclave and
   :new.clientedifesacve = relacionembarque.clientedifesacve and
   :new.tipoclientedifesacve = relacionembarque.tipoclientedifesacve and
   :new.empresacve = relacionembarque.empresacve and
   :new.situacionclientecve = relacionembarque.situacionclientecve and
   :new.tipoclientecve = relacionembarque.tipoclientecve and
   :new.tipoembarquecve = relacionembarque.tipoembarquecve;
if (
```

```
    /*  */

    numrows = 0
  )
  then
    raise_application_error(
      -20002,
      'Cannot INSERT almacenentradadistribucion because relacionembarque does not exist.'
    );
  end if;



end;
/

create trigger tU_almacenentradadistribucion after UPDATE on almacenentradadistribucion for each row

-- UPDATE trigger on almacenentradadistribucion
declare numrows INTEGER;
begin

  /* estatusdistribucion R/49 almacenentradadistribucion ON CHILD UPDATE RESTRICT */
  select count(*) into numrows
    from estatusdistribucion
    where
      /* :new.estatusdistribucioncve = estatusdistribucion.estatusdistribucioncve */
      :new.estatusdistribucioncve = estatusdistribucion.estatusdistribucioncve;
  if (
    /*  */

    numrows = 0
  )
  then
    raise_application_error(
      -20007,
      'Cannot UPDATE almacenentradadistribucion because estatusdistribucion does not exist.'
    );
  end if;


  /* clienteentrega R/48 almacenentradadistribucion ON CHILD UPDATE RESTRICT */
  select count(*) into numrows
    from clienteentrega
    where
      /* :new puntoentregacve = clienteentrega.puntoentregacve and
         :new.clientedifesacve = clienteentrega.clientedifesacve and
         :new.tipoclientedifesacve = clienteentrega.tipoclientedifesacve and
         :new.empresacve = clienteentrega.empresacve and
         :new.situacionclientecve = clienteentrega.situacionclientecve and
         :new.tipoclientecve = clienteentrega.tipoclientecve and
         :new.rutacve = clienteentrega.rutacve and
         :new.empleadocve = clienteentrega.empleadocve and
         :new.departamentocve = clienteentrega.departamentocve and
         :new.cargocve = clienteentrega.cargocve */
      :new.puntoentregacve = clienteentrega.puntoentregacve and
      :new.clientedifesacve = clienteentrega.clientedifesacve and
      :new.tipoclientedifesacve = clienteentrega.tipoclientedifesacve and
      :new.empresacve = clienteentrega.empresacve and
      :new.situacionclientecve = clienteentrega.situacionclientecve and
      :new.tipoclientecve = clienteentrega.tipoclientecve and
      :new.rutacve = clienteentrega.rutacve and
```

```
    :new.empleadocve = clienteentrega.empleadocve and
    :new.departamentocve = clienteentrega.departamentocve and
    :new.cargocve = clienteentrega.cargocve;
 if (
  /* */

  numrows = 0
 )
 then
  raise_application_error(
    -20007,
    'Cannot UPDATE almacenentradadistribucion because clienteentrega does not exist.'
  );
 end if;


 /* relacionembarque R/46 almacenentradadistribucion ON CHILD UPDATE RESTRICT */
 select count(*) into numrows
  from relacionembarque
  where
   /* :new.relacionembarqueclave = relacionembarque.relacionembarqueclave and
      :new.clientedifesacve = relacionembarque.clientedifesacve and
      :new.tipoclientedifesacve = relacionembarque.tipoclientedifesacve and
      :new.empresacve = relacionembarque.empresacve and
      :new.situacionclientecve = relacionembarque.situacionclientecve and
      :new.tipoclientecve = relacionembarque.tipoclientecve and
      :new.tipoembarquecve = relacionembarque.tipoembarquecve */
    :new.relacionembarqueclave = relacionembarque.relacionembarqueclave and
    :new.clientedifesacve = relacionembarque.clientedifesacve and
    :new.tipoclientedifesacve = relacionembarque.tipoclientedifesacve and
    :new.empresacve = relacionembarque.empresacve and
    :new.situacionclientecve = relacionembarque.situacionclientecve and
    :new.tipoclientecve = relacionembarque.tipoclientecve and
    :new.tipoembarquecve = relacionembarque.tipoembarquecve;
 if (
  /* */

  numrows = 0
 )
 then
  raise_application_error(
    -20007,
    'Cannot UPDATE almacenentradadistribucion because relacionembarque does not exist.'
  );
 end if;



end;
/

create trigger tl_almacenentradapublicacion after INSERT on almacenentradapublicacion for each row

-- INSERT trigger on almacenentradapublicacion
declare numrows INTEGER;
begin

   /* compra R/32 almacenentradapublicacion ON CHILD INSERT RESTRICT */
   select count(*) into numrows
    from compra
    where
     /* :new.compracve = compra.compracve and
```

```
      :new.empleadocve = compra.empleadocve and
      :new.departamentocve = compra.departamentocve and
      :new.cargocve = compra.cargocve and
      :new.proveedorcve = compra.proveedorcve and
      :new.empresacve = compra.empresacve */
    :new.compracve = compra.compracve and
    :new.empleadocve = compra.empleadocve and
    :new.departamentocve = compra.departamentocve and
    :new.cargocve = compra.cargocve and
    :new proveedorcve = compra.proveedorcve and
    :new.empresacve = compra.empresacve;
  if (
   /* */

   numrows = 0
  )
  then
   raise_application_error(
    -20002,
    'Cannot INSERT almacenentradapublicacion because compra does not exist.'
   );
  end if;


  /* edicion R/27 almacenentradapublicacion ON CHILD INSERT RESTRICT */
  select count(*) into numrows
   from edicion
   where
    /* :new.edicioncve = edicion.edicioncve and
       :new.publicacioncve = edicion.publicacioncve and
       :new.proveedorcve = edicion.proveedorcve and
       :new.empresacve = edicion.empresacve */
    :new.edicioncve = edicion.edicioncve and
    :new.publicacioncve = edicion.publicacioncve and
    :new.proveedorcve = edicion.proveedorcve and
    :new.empresacve = edicion.empresacve;
  if (
   /* */

   numrows = 0
  )
  then
   raise_application_error(
    -20002,
    'Cannot INSERT almacenentradapublicacion because edicion does not exist.'
   );
  end if;



end;
/

create trigger tU_almacenentradapublicacion after UPDATE on almacenentradapublicacion for each row

-- UPDATE trigger on almacenentradapublicacion
declare numrows INTEGER;
begin

 /* compra R/32 almacenentradapublicacion ON CHILD UPDATE RESTRICT */
 select count(*) into numrows
  from compra
```

```
  where
    /* :new.compracve = compra.compracve and
       :new.empleadocve = compra.empleadocve and
       :new.departamentocve = compra.departamentocve and
       :new.cargocve = compra.cargocve and
       :new.proveedorcve = compra.proveedorcve and
       :new.empresacve = compra.empresacve */
     :new.compracve = compra.compracve and
     :new.empleadocve = compra.empleadocve and
     :new.departamentocve = compra.departamentocve and
     :new.cargocve = compra.cargocve and
     :new.proveedorcve = compra.proveedorcve and
     :new.empresacve = compra.empresacve;
  if (
    /* */

    numrows = 0
  )
  then
    raise_application_error(
      -20007,
      'Cannot UPDATE almacenentradapublicacion because compra does not exist.'
    );
  end if;


  /* edicion R/27 almacenentradapublicacion ON CHILD UPDATE RESTRICT */
  select count(*) into numrows
    from edicion
    where
      /* :new.edicioncve = edicion.edicioncve and
         :new.publicacioncve = edicion.publicacioncve and
         :new.proveedorcve = edicion.proveedorcve and
         :new.empresacve = edicion.empresacve */
      :new.edicioncve = edicion.edicioncve and
      :new.publicacioncve = edicion.publicacioncve and
      :new.proveedorcve = edicion.proveedorcve and
      :new.empresacve = edicion.empresacve;
  if (
    /* */

    numrows = 0
  )
  then
    raise_application_error(
      -20007,
      'Cannot UPDATE almacenentradapublicacion because edicion does not exist.'
    );
  end if;



end;
/

create trigger tl_bitacoraevento after INSERT on bitacoraevento for each row

-- INSERT trigger on bitacoraevento
declare numrows INTEGER;
begin

    /* tipoevento R/63 bitacoraevento ON CHILD INSERT RESTRICT */
```

```
  select count(*) into numrows
    from tipoevento
    where
      /* :new.tipoeventocve = tipoevento.tipoeventocve */
      :new.tipoeventocve = tipoevento.tipoeventocve;
  if (
    /* */

    numrows = 0
  )
  then
    raise_application_error(
      -20002,
      'Cannot INSERT bitacoraevento because tipoevento does not exist.'
    );
  end if;


  /* ruta R/62 bitacoraevento ON CHILD INSERT RESTRICT */
  select count(*) into numrows
    from ruta
    where
      /* :new.rutacve = ruta.rutacve and
         :new.empleadocve = ruta.empleadocve and
         :new.departamentocve = ruta.departamentocve and
         :new.cargocve = ruta.cargocve */
      :new.rutacve = ruta.rutacve and
      :new.empleadocve = ruta.empleadocve and
      :new.departamentocve = ruta.departamentocve and
      :new.cargocve = ruta.cargocve;
  if (
    /* */

    numrows = 0
  )
  then
    raise_application_error(
      -20002,
      'Cannot INSERT bitacoraevento because ruta does not exist.'
    );
  end if;



end;
/

create trigger tU_bitacoraevento after UPDATE on bitacoraevento for each row

-- UPDATE trigger on bitacoraevento
declare numrows INTEGER;
begin

  /* tipoevento R/63 bitacoraevento ON CHILD UPDATE RESTRICT */
  select count(*) into numrows
    from tipoevento
    where
      /* :new.tipoeventocve = tipoevento.tipoeventocve */
      :new.tipoeventocve = tipoevento.tipoeventocve;
  if (
    /* */
```

```
  numrows = 0
)
then
  raise_application_error(
    -20007,
    'Cannot UPDATE bitacoraevento because tipoevento does not exist.'
  );
end if;


/* ruta R/62 bitacoraevento ON CHILD UPDATE RESTRICT */
select count(*) into numrows
  from ruta
  where
    /* :new.rutacve = ruta.rutacve and
       :new.empleadocve = ruta.empleadocve and
       :new.departamentocve = ruta.departamentocve and
       :new.cargocve = ruta.cargocve */
    :new.rutacve = ruta.rutacve and
    :new.empleadocve = ruta.empleadocve and
    :new.departamentocve = ruta.departamentocve and
    :new.cargocve = ruta.cargocve;
if (
  /* */

  numrows = 0
)
then
  raise_application_error(
    -20007,
    'Cannot UPDATE bitacoraevento because ruta does not exist.'
  );
end if;



end;
/

create trigger tD_cargo after DELETE on cargo for each row

-- DELETE trigger on cargo
declare numrows INTEGER;
begin

   /* cargo R/6 empleado ON PARENT DELETE RESTRICT */
   select count(*) into numrows
     from empleado
     where
       /* empleado.cargocve = :old.cargocve */
       empleado.cargocve = :old.cargocve;
   if (numrows > 0)
   then
     raise_application_error(
       -20001,
       'Cannot DELETE cargo because empleado exists.'
     );
   end if;

end;
/
```

```
create trigger tU_cargo after UPDATE on cargo for each row

-- UPDATE trigger on cargo
declare numrows INTEGER;
begin

 /* cargo R/6 empleado ON PARENT UPDATE RESTRICT */
 if
  /* :old.cargocve <> :new.cargocve */
  :old.cargocve <> :new.cargocve
 then
  select count(*) into numrows
   from empleado
   where
    /* empleado.cargocve = :old.cargocve */
    empleado.cargocve = :old.cargocve;
  if (numrows > 0)
  then
   raise_application_error(
    -20005,
    'Cannot UPDATE cargo because empleado exists.'
   );
  end if;
 end if;

end;
/

create trigger tD_clientedifesa after DELETE on clientedifesa for each row

-- DELETE trigger on clientedifesa
declare numrows INTEGER;
begin

  /* clientedifesa R/53 distribucioncosto ON PARENT DELETE RESTRICT */
  select count(*) into numrows
   from distribucioncosto
   where
    /* distribucioncosto.clientedifesacve = :old.clientedifesacve and
       distribucioncosto.tipoclientedifesacve = :old.tipoclientedifesacve and
       distribucioncosto.empresacve = :old.empresacve and
       distribucioncosto.situacionclientecve = :old.situacionclientecve and
       distribucioncosto.tipoclientecve = :old.tipoclientecve */
    distribucioncosto.clientedifesacve = :old.clientedifesacve and
    distribucioncosto.tipoclientedifesacve = :old.tipoclientedifesacve and
    distribucioncosto.empresacve = :old.empresacve and
    distribucioncosto.situacionclientecve = :old.situacionclientecve and
    distribucioncosto.tipoclientecve = :old.tipoclientecve;
  if (numrows > 0)
  then
   raise_application_error(
    -20001,
    'Cannot DELETE clientedifesa because distribucioncosto exists.'
   );
  end if;


  /* clientedifesa R/45 relacionembarque ON PARENT DELETE RESTRICT */
  select count(*) into numrows
   from relacionembarque
   where
    /* relacionembarque.clientedifesacve = :old.clientedifesacve and
```

```
      relacionembarque.tipoclientedifesacve = :old.tipoclientedifesacve and
      relacionembarque.empresacve = :old.empresacve and
      relacionembarque.situacionclientecve = :old.situacionclientecve and
      relacionembarque.tipoclientecve = :old.tipoclientecve */
    relacionembarque.clientedifesacve = :old.clientedifesacve and
    relacionembarque.tipoclientedifesacve = :old.tipoclientedifesacve and
    relacionembarque.empresacve = :old.empresacve and
    relacionembarque.situacionclientecve = :old.situacionclientecve and
    relacionembarque.tipoclientecve = :old.tipoclientecve;
if (numrows > 0)
then
  raise_application_error(
    -20001,
    'Cannot DELETE clientedifesa because relacionembarque exists.'
  );
end if;


/* clientedifesa R/39 devolucion ON PARENT DELETE RESTRICT */
select count(*) into numrows
  from devolucion
  where
    /* devolucion.clientedifesacve = :old.clientedifesacve and
       devolucion.tipoclientedifesacve = :old.tipoclientedifesacve and
       devolucion.empresacve = :old.empresacve and
       devolucion.situacionclientecve = :old.situacionclientecve and
       devolucion.tipoclientecve = :old.tipoclientecve */
    devolucion.clientedifesacve = :old.clientedifesacve and
    devolucion.tipoclientedifesacve = :old.tipoclientedifesacve and
    devolucion.empresacve = :old.empresacve and
    devolucion.situacionclientecve = :old.situacionclientecve and
    devolucion.tipoclientecve = :old.tipoclientecve;
if (numrows > 0)
then
  raise_applieation_error(
    -20001,
    'Cannot DELETE clientedifesa because devolucion exists.'
  );
end if;


/* clientedifesa R/33 facturapublicacion ON PARENT DELETE RESTRICT */
select count(*) into numrows
  from facturapublicacion
  where
    /* facturapublicacion.clientedifesacve = :old.clientedifesacve and
       facturapublicacion.tipoclientedifesacve = :old.tipoclientedifesacve and
       facturapublicacion.empresacve = :old.empresacve and
       facturapublicacion.situacionclientecve = :old.situacionclientecve and
       facturapublicacion.tipoclientecve = :old.tipoclientecve */
    facturapublicacion.clientedifesacve = :old.clientedifesacve and
    facturapublicacion.tipoclientedifesacve = :old.tipoclientedifesacve and
    facturapublicacion.empresacve = :old.empresacve and
    facturapublicacion.situacionclientecve = :old.situacionclientecve and
    facturapublicacion.tipoclientecve = :old.tipoclientecve;
if (numrows > 0)
then
  raise_application_error(
    -20001,
    'Cannot DELETE clientedifesa because facturapublicacion exists.'
  );
end if;
```

```
/* clientedifesa R/19 descuento ON PARENT DELETE RESTRICT */
select count(*) into numrows
  from descuento
  where
   /*  descuento.clientedifesacve = :old.clientedifesacve and
      descuento.tipoclientedifesacve = :old.tipoclientedifesacve and
      descuento.empresacve = :old.empresacve and
      descuento.situacionclientecve = :old.situacionclientecve and
      descuento.tipoclientecve = :old.tipoclientecve */
    descuento.clientedifesacve = :old.clientedifesacve and
    descuento.tipoclientedifesacve = :old.tipoclientedifesacve and
    descuento.empresacve = :old.empresacve and
    descuento.situacionclientecve = :old.situacionclientecve and
    descuento.tipoclientecve = :old.tipoclientecve;
if (numrows > 0)
then
  raise_application_error(
    -20001,
    'Cannot DELETE clientedifesa because descuento exists.'
  );
end if;


/* clientedifesa R/15 clienteentrega ON PARENT DELETE RESTRICT */
select count(*) into numrows
  from clienteentrega
  where
   /*  clienteentrega.clientedifesacve = :old.clientedifesacve and
      clienteentrega.tipoclientedifesacve = :old.tipoclientedifesacve and
      clienteentrega.empresacve = :old.empresacve and
      clienteentrega.situacionclientecve = :old.situacionclientecve and
      clienteentrega.tipoclientecve = :old.tipoclientecve */
    clienteentrega.clientedifesacve = :old.clientedifesacve and
    clienteentrega.tipoclientedifesacve = :old.tipoclientedifesacve and
    clienteentrega.empresacve = :old.empresacve and
    clienteentrega.situacionclientecve = :old.situacionclientecve and
    clienteentrega.tipoclientecve = :old.tipoclientecve;
if (numrows > 0)
then
  raise_application_error(
    -20001,
    'Cannot DELETE clientedifesa because clienteentrega exists.'
  );
end if;



end;
/

create trigger tI_clientedifesa after INSERT on clientedifesa for each row

-- INSERT trigger on clientedifesa
declare numrows INTEGER;
begin

   /* tipocliente R/13 clientedifesa ON CHILD INSERT RESTRICT */
   select count(*) into numrows
     from tipocliente
     where
```

```
    /* :new.tipoclientecve = tipocliente.tipoclientecve */
    :new.tipoclientecve = tipocliente.tipoclientecve;
  if (
   /*  */

   numrows = 0
  )
  then
   raise_application_error(
    -20002,
    'Cannot INSERT clientedifesa because tipocliente does not exist.'
   );
  end if;


  /* situacioncliente R/12 clientedifesa ON CHILD INSERT RESTRICT */
  select count(*) into numrows
   from situacioncliente
   where
    /* :new.situacionclientecve = situacioncliente.situacionclientecve */
    :new.situacionclientecve = situacioncliente.situacionclientecve;
  if (
   /*  */

   numrows = 0
  )
  then
   raise_application_error(
    -20002,
    'Cannot INSERT clientedifesa because situacioncliente does not exist.'
   );
  end if;


  /* empresa R/11 clientedifesa ON CHILD INSERT RESTRICT */
  select count(*) into numrows
   from empresa
   where
    /* :new.empresacve = empresa.empresacve */
    :new.empresacve = empresa.empresacve;
  if (
   /*  */

   numrows = 0
  )
  then
   raise_application_error(
    -20002,
    'Cannot INSERT clientedifesa because empresa does not exist.'
   );
  end if;



end;
/

create trigger tU_clientedifesa after UPDATE on clientedifesa for each row

-- UPDATE trigger on clientedifesa
declare numrows INTEGER;
begin
```

```
/* clientedifesa R/53 distribucioncosto ON PARENT UPDATE RESTRICT */
if
 /* :old.clientedifesacve <> :new.clientedifesacve or
    :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
    :old.empresacve <> :new.empresacve or
    :old.situacionclientecve <> :new.situacionclientecve or
    :old.tipoclientecve <> :new.tipoclientecve */
  :old.clientedifesacve <> :new.clientedifesacve or
  :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
  :old.empresacve <> :new.empresacve or
  :old.situacionclientecve <> :new.situacionclientecve or
  :old.tipoclientecve <> :new.tipoclientecve
then
 select count(*) into numrows
  from distribucioncosto
  where
   /* distribucioncosto.clientedifesacve = :old.clientedifesacve and
      distribucioncosto.tipoclientedifesacve = :old.tipoclientedifesacve and
      distribucioncosto.empresacve = :old.empresacve and
      distribucioncosto.situacionclientecve = :old.situacionclientecve and
      distribucioncosto.tipoclientecve = :old.tipoclientecve */
    distribucioncosto.clientedifesacve = :old.clientedifesacve and
    distribucioncosto.tipoclientedifesacve = :old.tipoclientedifesacve and
    distribucioncosto.empresacve = :old.empresacve and
    distribucioncosto.situacionclientecve = :old.situacionclientecve and
    distribucioncosto.tipoclientecve = :old.tipoclientecve;
 if (numrows > 0)
 then
  raise_application_error(
   -20005,
   'Cannot UPDATE clientedifesa because distribucioncosto exists.'
  );
 end if;
end if;


/* clientedifesa R/45 relacionembarque ON PARENT UPDATE RESTRICT */
if
 /* :old.clientedifesacve <> :new.clientedifesacve or
    :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
    :old.empresacve <> :new.empresacve or
    :old.situacionclientecve <> :new.situacionclientecve or
    :old.tipoclientecve <> :new.tipoclientecve */
  :old.clientedifesacve <> :new.clientedifesacve or
  :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
  :old.empresacve <> :new.empresacve or
  :old.situacionclientecve <> :new.situacionclientecve or
  :old.tipoclientecve <> :new.tipoclientecve
then
  select count(*) into numrows
   from relacionembarque
   where
    /* relacionembarque.clientedifesacve = :old.clientedifesacve and
       relacionembarque.tipoclientedifesacve = :old.tipoclientedifesacve and
       relacionembarque.empresacve = :old.empresacve and
       relacionembarque.situacionclientecve = :old.situacionclientecve and
       relacionembarque.tipoclientecve = :old.tipoclientecve */
     relacionembarque.clientedifesacve = :old.clientedifesacve and
     relacionembarque.tipoclientedifesacve = :old.tipoclientedifesacve and
     relacionembarque.empresacve = :old.empresacve and
     relacionembarque.situacionclientecve = :old.situacionclientecve and
```

```
    relacionembarque.tipoclientecve = :old.tipoclientecve;
  if (numrows > 0)
  then
   raise_application_error(
    -20005,
    'Cannot UPDATE clientedifesa because relacionembarque exists.'
   );
  end if;
end if;


/* clientedifesa R/39 devolucion ON PARENT UPDATE RESTRICT */
if
  /* :old.clientedifesacve <> :new.clientedifesacve or
    :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
    :old.empresacve <> :new.empresacve or
    :old.situacionclientecve <> :new.situacionclientecve or
    :old.tipoclientecve <> :new.tipoclientecve */
  :old.clientedifesacve <> :new.clientedifesacve or
  :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
  :old.empresacve <> :new.empresacve or
  :old.situacionclientecve <> :new.situacionclientecve or
  :old.tipoclientecve <> :new.tipoclientecve
then
  select count(*) into numrows
    from devolucion
    where
     /*  devolucion.clientedifesacve = :old.clientedifesacve and
        devolucion.tipoclientedifesacve = :old.tipoclientedifesacve and
        devolucion.empresacve = :old.empresacve and
        devolucion.situacionclientecve = :old.situacionclientecve and
        devolucion.tipoclientecve = :old.tipoclientecve */
     devolucion.clientedifesacve = :old.clientedifesacve and
     devolucion.tipoclientedifesacve = :old.tipoclientedifesacve and
     devolucion.empresacve = :old.empresacve and
     devolucion.situacionclientecve = :old.situacionclientecve and
     devolucion.tipoclientecve = :old.tipoclientecve;
  if (numrows > 0)
  then
   raise_application_error(
    -20005,
    'Cannot UPDATE clientedifesa because devolucion exists.'
   );
  end if;
end if;


/* clientedifesa R/33 facturapublicacion ON PARENT UPDATE RESTRICT */
if
  /* :old.clientedifesacve <> :new.clientedifesacve or
    :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
    :old.empresacve <> :new.empresacve or
    :old.situacionclientecve <> :new.situacionclientecve or
    :old.tipoclientecve <> :new.tipoclientecve */
  :old.clientedifesacve <> :new.clientedifesacve or
  :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
  :old.empresacve <> :new.empresacve or
  :old.situacionclientecve <> :new.situacionclientecve or
  :old.tipoclientecve <> :new.tipoclientecve
then
  select count(*) into numrows
    from facturapublicacion
```

```
   where
    /*  facturapublicacion.clientedifesacve = :old.clientedifesacve and
        facturapublicacion.tipoclientedifesacve = :old.tipoclientedifesacve and
        facturapublicacion.empresacve = :old.empresacve and
        facturapublicacion.situacionclientecve = :old.situacionclientecve and
        facturapublicacion.tipoclientecve = :old.tipoclientecve */
     facturapublicacion.clientedifesacve = :old.clientedifesacve and
     facturapublicacion.tipoclientedifesacve = :old.tipoclientedifesacve and
     facturapublicacion.empresacve = :old.empresacve and
     facturapublicacion.situacionclientecve = :old.situacionclientecve and
     facturapublicacion.tipoclientecve = :old.tipoclientecve;
  if (numrows > 0)
  then
   raise_application_error(
    -20005,
    'Cannot UPDATE clientedifesa because facturapublicacion exists.'
   );
  end if;
 end if;


/* clientedifesa R/19 descuento ON PARENT UPDATE RESTRICT */
if
  /* :old.clientedifesacve <> :new.clientedifesacve or
     :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
     :old.empresacve <> :new.empresacve or
     :old.situacionclientecve <> :new.situacionclientecve or
     :old.tipoclientecve <> :new.tipoclientecve */
  :old.clientedifesacve <> :new.clientedifesacve or
  :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
  :old.empresacve <> :new.empresacve or
  :old.situacionclientecve <> :new.situacionclientecve or
  :old.tipoclientecve <> :new.tipoclientecve
then
  selcct count(*) into numrows
   from descuento
   where
    /*  descuento.clientedifesacve = :old.clientedifesacve and
        descuento.tipoclientedifesacve = :old.tipoclientedifesacve and
        descuento.empresacve = :old.empresacve and
        descuento.situacionclientecve = :old.situacionclientecve and
        descuento.tipoclientecve = :old.tipoclientecve */
     descuento.clientedifesacve = :old.clientedifesacve and
     descuento.tipoclientedifesacve = :old.tipoclientedifesacve and
     descuento.empresacve = :old.empresacve and
     descuento.situacionclientecve = :old.situacionclientecve and
     descuento.tipoclientecve = :old.tipoclientecve;
  if (numrows > 0)
  then
   raise_application_error(
    -20005,
    'Cannot UPDATE clientedifesa because descuento exists.'
   );
  end if;
 end if;


/* clientedifesa R/15 elienteentrega ON PARENT UPDATE RESTRICT */
if
  /* :old.clientedifesacve <> :new.clientedifesacve or
     :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
     :old.empresacve <> :new.empresacve or
```

```
    :old.situacionclientecve <> :new.situacionclientecve or
    :old.tipoclientecve <> :new.tipoclientecve */
  :old.clientedifesacve <> :new.clientedifesacve or
  :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
  :old.empresacve <> :new.empresacve or
  :old.situacionclientecve <> :new.situacionclientecve or
  :old.tipoclientecve <> :new.tipoclientecve
then
  select count(*) into numrows
    from clienteentrega
    where
      /*  clienteentrega.clientedifesacve = :old.clientedifesacve and
          clienteentrega.tipoclientedifesacve = :old.tipoclientedifesacve and
          clienteentrega.empresacve = :old.empresacve and
          clienteentrega.situacionclientecve = :old.situacionclientecve and
          clienteentrega.tipoclientecve = :old.tipoclientecve */
      clienteentrega.clientedifesacve = :old.clientedifesacve and
      clienteentrega.tipoclientedifesacve = :old.tipoclientedifesacve and
      clienteentrega.empresacve = :old.empresacve and
      clienteentrega.situacionclientecve = :old.situacionclientecve and
      clienteentrega.tipoclientecve = :old.tipoclientecve;
  if (numrows > 0)
  then
    raise_application_error(
      -20005,
      'Cannot UPDATE clientedifesa because clienteentrega exists.'
    );
  end if;
end if;


/* tipocliente R/13 clientedifesa ON CHILD UPDATE RESTRICT */
select count(*) into numrows
  from tipocliente
  where
    /* :new.tipoclientecve = tipocliente.tipoclientecve */
    :new.tipoclientecve = tipocliente.tipoclientecve;
if (
  /*  */

  numrows = 0
)
then
  raise_application_error(
    -20007,
    'Cannot UPDATE clientedifesa because tipocliente does not exist.'
  );
end if;


/* situacioncliente R/12 clientedifesa ON CHILD UPDATE RESTRICT */
select count(*) into numrows
  from situacioncliente
  where
    /* :new.situacionclientecve = situacioncliente.situacionclientecve */
    :new.situacionclientecve = situacioncliente.situacionclientecve;
if (
  /*  */

  numrows = 0
)
then
```

```
  raise_application_error(
    -20007,
    'Cannot UPDATE clientedifesa because situacioncliente does not exist.'
  );
 end if;


 /* empresa R/11 clientedifesa ON CHILD UPDATE RESTRICT */
 select count(*) into numrows
  from empresa
  where
    /* :new.empresacve = empresa.empresacve */
    :new.empresacve = empresa.empresacve;
 if (
  /*  */

  numrows = 0
 )
 then
  raise_application_error(
    -20007,
    'Cannot UPDATE clientedifesa because empresa does not exist.'
  );
 end if;



end;
/

create trigger tD_clienteentrega after DELETE on clienteentrega for each row

-- DELETE trigger on clienteentrega
declare numrows INTEGER;
begin

   /* clienteentrega R/48 almacenentradadistribucion ON PARENT DELETE RESTRICT */
   select count(*) into numrows
    from almacenentradadistribucion
    where
     /*  almacenentradadistribucion.puntoentregacve = :old.puntoentregacve and
        almacenentradadistribucion.clientedifesacve = :old.clientedifesacve and
        almacenentradadistribucion.tipoclientedifesacve = :old.tipoclientedifesacve and
        almacenentradadistribucion.empresacve = :old.empresacve and
        almacenentradadistribucion.situacionclientecve = :old.situacionclientecve and
        almacenentradadistribucion.tipoclientecve = :old.tipoclientecve and
        almacenentradadistribucion.rutacve = :old.rutacve and
        almacenentradadistribucion.empleadocve = :old.empleadocve and
        almacenentradadistribucion.departamentocve = :old.departamentocve and
        almacenentradadistribucion.cargocve = :old.cargocve */
      almacenentradadistribucion.puntoentregacve = :old.puntoentregacve and
      almacenentradadistribucion.clientedifesacve = :old.clientedifesacve and
      almacenentradadistribucion.tipoclientedifesacve = :old.tipoclientedifesacve and
      almacenentradadistribucion.empresacve = :old.empresacve and
      almacenentradadistribucion.situacionclientecve = :old.situacionclientecve and
      almacenentradadistribucion.tipoclientecve = :old.tipoclientecve and
      almacenentradadistribucion.rutacve = :old.rutacve and
      almacenentradadistribucion.empleadocve = :old.empleadocve and
      almacenentradadistribucion.departamentocve = :old.departamentocve and
      almacenentradadistribucion.cargocve = :old.cargocve;
   if (numrows > 0)
   then
```

```
    raise_application_error(
      -20001,
      'Cannot DELETE clienteentrega because almacenentradadistribucion exists.'
    );
   end if;



end;
/

create trigger tI_clienteentrega after INSERT on clienteentrega for each row

-- INSERT trigger on clienteentrega
declare numrows INTEGER;
begin

   /* clientedifesa R/15 clienteentrega ON CHILD INSERT RESTRICT */
   select count(*) into numrows
    from clientedifesa
    where
     /* :new.clientedifesacve = clientedifesa.clientedifesacve and
        :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
        :new.empresacve = clientedifesa.empresacve and
        :new.situacionclientecve = clientedifesa.situacionclientecve and
        :new.tipoclientecve = clientedifesa.tipoclientecve */
      :new.clientedifesacve = clientedifesa.clientedifesacve and
      :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
      :new.empresacve = clientedifesa.empresacve and
      :new.situacionclientecve = clientedifesa.situacionclientecve and
      :new.tipoclientecve = clientedifesa.tipoclientecve;
   if (
    /* */

     numrows = 0
   )
   then
    raise_application_error(
      -20002,
      'Cannot INSERT clienteentrega because clientedifesa does not exist.'
    );
   end if;


   /* puntoentrega R/14 clienteentrega ON CHILD INSERT RESTRICT */
   select count(*) into numrows
    from puntoentrega
    where
     /* :new.puntoentregacve = puntoentrega.puntoentregacve and
        :new.rutacve = puntoentrega.rutacve and
        :new.empleadocve = puntoentrega.empleadocve and
        :new.departamentocve = puntoentrega.departamentocve and
        :new.cargocve = puntoentrega.cargocve */
      :new.puntoentregacve = puntoentrega.puntoentregacve and
      :new.rutacve = puntoentrega.rutacve and
      :new.empleadocve = puntoentrega.empleadocve and
      :new.departamentocve = puntoentrega.departamentocve and
      :new.cargocve = puntoentrega.cargocve;
   if (
    /* */

     numrows = 0
```

```
  )
  then
   raise_application_error(
    -20002,
    'Cannot INSERT clienteentrega because puntoentrega does not exist.'
   );
  end if;



end;
/

create trigger tU_clienteentrega after UPDATE on clienteentrega for each row

-- UPDATE trigger on clienteentrega
declare numrows INTEGER;
begin

 /* clienteentrega R/48 almacenentradadistribucion ON PARENT UPDATE RESTRICT */
 if
  /* :old.puntoentregacve <> :new.puntoentregacve or
     :old.clientedifesacve <> :new.clientedifesacve or
     :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
     :old.empresacve <> :new.empresacve or
     :old.situacionclientecve <> :new.situacionclientecve or
     :old.tipoclientecve <> :new.tipoclientecve or
     :old.rutacve <> :new.rutacve or
     :old.empleadocve <> :new.empleadocve or
     :old.departamentocve <> :new.departamentocve or
     :old.cargocve <> :new.cargocve */
  :old.puntoentregacve <> :new.puntoentregacve or
  :old.clientedifesacve <> :new.clientedifesacve or
  :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
  :old.empresacve <> :new.empresacve or
  :old.situacionclientecve <> :new.situacionclientecve or
  :old.tipoclientecve <> :new.tipoclientecve or
  :old.rutacve <> :new.rutacve or
  :old.empleadocve <> :new.empleadocve or
  :old.departamentocve <> :new.departamentocve or
  :old.cargocve <> :new.cargocve
 then
  select count(*) into numrows
   from almacenentradadistribucion
   where
    /* almacenentradadistribucion.puntoentregacve = :old.puntoentregacve and
       almacenentradadistribucion.clientedifesacve = :old.clientedifesacve and
       almacenentradadistribucion.tipoclientedifesacve = :old.tipoclientedifesacve and
       almacenentradadistribucion.empresacve = :old.empresacve and
       almacenentradadistribucion.situacionclientecve = :old.situacionclientecve and
       almacenentradadistribucion.tipoclientecve = :old.tipoclientecve and
       almacenentradadistribucion.rutacve = :old.rutacve and
       almacenentradadistribucion.empleadocve = :old.empleadocve and
       almacenentradadistribucion.departamentocve = :old.departamentocve and
       almacenentradadistribucion.cargocve = :old.cargocve */
     almacenentradadistribucion.puntoentregacve = :old.puntoentregacve and
     almacenentradadistribucion.clientedifesacve = :old.clientedifesacve and
     almacenentradadistribucion.tipoclientedifesacve = :old.tipoclientedifesacve and
     almacenentradadistribucion.empresacve = :old.empresacve and
     almacenentradadistribucion.situacionclientecve = :old.situacionclientecve and
     almacenentradadistribucion.tipoclientecve = :old.tipoclientecve and
     almacenentradadistribucion.rutacve = :old.rutacve and
```

```
    almacenentradadistribucion.empleadocve = :old.empleadocve and
    almacenentradadistribucion.departamentocve = :old.departamentocve and
    almacenentradadistribucion.cargocve = :old.cargocve;
  if (numrows > 0)
  then
   raise_application_error(
    -20005,
    'Cannot UPDATE clienteentrega because almacenentradadistribucion exists.'
   );
  end if;
end if;


/* clientedifesa R/15 clienteentrega ON CHILD UPDATE RESTRICT */
select count(*) into numrows
 from clientedifesa
 where
  /* :new.clientedifesacve = clientedifesa.clientedifesacve and
     :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
     :new.empresacve = clientedifesa.empresacve and
     :new.situacionclientecve = clientedifesa.situacionclientecve and
     :new.tipoclientecve = clientedifesa.tipoclientecve */
   :new.clientedifesacve = clientedifesa.clientedifesacve and
   :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
   :new.empresacve = clientedifesa.empresacve and
   :new.situacionclientecve = clientedifesa.situacionclientecve and
   :new.tipoclientecve = clientedifesa.tipoclientecve;
if (
 /* */

 numrows = 0
)
then
 raise_application_error(
  -20007,
  'Cannot UPDATE clienteentrega because clientedifesa does not exist.'
 );
end if;


/* puntocntrega R/14 clienteentrega ON CHILD UPDATE RESTRICT */
select count(*) into numrows
 from puntoentrega
 where
  /* :new.puntoentregacve = puntoentrega.puntoentregacve and
     :new.rutacve = puntoentrega.rutacve and
     :new.empleadocve = puntoentrega.empleadocve and
     :new.departamentocve = puntoentrega.departamentocve and
     :new.cargocve = puntoentrega.cargocve */
   :new.puntoentregacve = puntoentrega.puntoentregacve and
   :new.rutacve = puntoentrega.rutacve and
   :new.empleadocve = puntoentrega.empleadocve and
   :new.departamentocve = puntoentrega.departamentocve and
   :new.cargocve = puntoentrega.cargocve;
if (
 /* */

 numrows = 0
)
then
 raise_application_error(
  -20007,
```

```
    'Cannot UPDATE clienteentrega because puntoentrega does not exist.'
  );
 end if;



end;
/

create trigger tD_compra after DELETE on compra for each row

-- DELETE trigger on compra
declare numrows INTEGER;
begin

  /* compra R/32 almacenentradapublicacion ON PARENT DELETE RESTRICT */
  select count(*) into numrows
    from almacenentradapublicacion
    where
      /*  almacenentradapublicacion.compracve = :old.compracve and
          almacenentradapublicacion.empleadocve = :old.empleadocve and
          almacenentradapublicacion.departamentocve = :old.departamentocve and
          almacenentradapublicacion.cargocve = :old.cargocve and
          almacenentradapublicacion.proveedorcve = :old.proveedorcve and
          almacenentradapublicacion.empresacve = :old.empresacve */
      almacenentradapublicacion.compracve = :old.compracve and
      almacenentradapublicacion.empleadocve = :old.empleadocve and
      almacenentradapublicacion.departamentocve = :old.departamentocve and
      almacenentradapublicacion.cargocve = :old.cargocve and
      almacenentradapublicacion.proveedorcve = :old.proveedorcve and
      almacenentradapublicacion.empresacve = :old.empresacve;
  if (numrows > 0)
  then
    raise_application_error(
      -20001,
      'Cannot DELETE compra because almacenentradapublicacion exists.'
    );
  end if;


  /* compra R/31 comprapago ON PARENT DELETE RESTRICT */
  select count(*) into numrows
    from comprapago
    where
      /*  comprapago.compracve = :old.compracve and
          comprapago.empleadocve = :old.empleadocve and
          comprapago.departamentocve = :old.departamentocve and
          comprapago.cargocve = :old.cargocve and
          comprapago.proveedorcve = :old.proveedorcve and
          comprapago.empresacve = :old.empresacve */
      comprapago.compracve = :old.compracve and
      comprapago.empleadocve = :old.empleadocve and
      comprapago.departamentocve = :old.departamentocve and
      comprapago.cargocve = :old.cargocve and
      comprapago.proveedorcve = :old.proveedorcve and
      comprapago.empresacve = :old.empresacve;
  if (numrows > 0)
  then
    raise_application_error(
      -20001,
      'Cannot DELETE compra because comprapago exists.'
    );
```

```
  end if;


end;
/

create trigger tI_compra after INSERT on compra for each row

-- INSERT trigger on compra
declare numrows INTEGER;
begin

  /* proveedor R/22 compra ON CHILD INSERT RESTRICT */
  select count(*) into numrows
    from proveedor
    where
     /* :new.proveedorcve = proveedor.proveedorcve and
        :new.empresacve = proveedor.empresacve */
      :new.proveedorcve = proveedor.proveedorcve and
      :new.empresacve = proveedor.empresacve;
  if (
   /* */

    numrows = 0
  )
  then
    raise_application_error(
      -20002,
      'Cannot INSERT compra because proveedor does not exist.'
    );
  end if;


  /* empleado R/21 compra ON CHILD INSERT RESTRICT */
  select count(*) into numrows
    from empleado
    where
     /* :new.empleadocve = empleado.empleadocve and
        :new.departamentocve = empleado.departamentocve and
        :new.cargocve = empleado.cargocve */
      :new.empleadocve = empleado.empleadocve and
      :new.departamentocve = empleado.departamentocve and
      :new.cargocve = empleado.cargocve;
  if (
   /* */

    numrows = 0
  )
  then
    raise_application_error(
      -20002,
      'Cannot INSERT compra because empleado does not exist.'
    );
  end if;



end;
/

create trigger tU_compra after UPDATE on compra for each row
```

```
-- UPDATE trigger on compra
declare numrows INTEGER;
begin

 /* compra R/32 almacenentradapublicacion ON PARENT UPDATE RESTRICT */
 if
  /* :old.compracve <> :new.compracve or
    :old.empleadocve <> :new.empleadocve or
    :old.departamentocve <> :new.departamentocve or
    :old.cargocve <> :new.cargocve or
    :old.proveedorcve <> :new.proveedorcve or
    :old.empresacve <> :new.empresacve */
  :old.compracve <> :new.compracve or
  :old.empleadocve <> :new.empleadocve or
  :old.departamentocve <> :new.departamentocve or
  :old.cargocve <> :new.cargocve or
  :old.proveedorcve <> :new.proveedorcve or
  :old.empresacve <> :new.empresacve
 then
  select count(*) into numrows
   from almacenentradapublicacion
   where
    /* almacenentradapublicacion.compracve = :old.compracve and
       almacenentradapublicacion.empleadocve = :old.empleadocve and
       almacenentradapublicacion.departamentocve = :old.departamentocve and
       almacenentradapublicacion.cargocve = :old.cargocve and
       almacenentradapublicacion.proveedorcve = :old.proveedorcve and
       almacenentradapublicacion.empresacve = :old.empresacve */
    almacenentradapublicacion.compracve = :old.compracve and
    almacenentradapublicacion.empleadocve = :old.empleadocve and
    almacenentradapublicacion.departamentocve = :old.departamentocve and
    almacenentradapublicacion.cargocve = :old.cargocve and
    almacenentradapublicacion.proveedorcve = :old.proveedorcve and
    almacenentradapublicacion.empresacve = :old.empresacve;
  if (numrows > 0)
  then
   raise_application_error(
    -20005,
    'Cannot UPDATE compra because almacenentradapublicacion exists.'
   );
  end if;
 end if;


 /* compra R/31 comprapago ON PARENT UPDATE RESTRICT */
 if
  /* :old.compracve <> :new.compracve or
    :old.empleadocve <> :new.empleadocve or
    :old.departamentocve <> :new.departamentocve or
    :old.cargocve <> :new.cargocve or
    :old.proveedorcve <> :new.proveedorcve or
    :old.empresacve <> :new.empresacve */
  :old.compracve <> :new.compracve or
  :old.empleadocve <> :new.empleadocve or
  :old.departamentocve <> :new.departamentocve or
  :old.cargocve <> :new.cargocve or
  :old.proveedorcve <> :new.proveedorcve or
  :old.empresacve <> :new.empresacve
 then
  select count(*) into numrows
   from comprapago
```

```
   where
    /*  comprapago.compracve = :old.compracve and
        comprapago.empleadocve = :old.empleadocve and
        comprapago.departamentocve = :old.departamentocve and
        comprapago.cargocve = :old.cargocve and
        comprapago.proveedorcve = :old.proveedorcve and
        comprapago.empresacve = :old.empresacve */
     comprapago.compracve = :old.compracve and
     comprapago.empleadocve = :old.empleadocve and
     comprapago.departamentocve = :old.departamentocve and
     comprapago.cargocve = :old.cargocve and
     comprapago.proveedorcve = :old.proveedorcve and
     comprapago.empresacve = :old.empresacve;
  if (numrows > 0)
  then
   raise_application_error(
    -20005,
    'Cannot UPDATE compra because comprapago exists.'
   );
  end if;
end if;


/* proveedor R/22 compra ON CHILD UPDATE RESTRICT */
select count(*) into numrows
  from proveedor
  where
    /* :new.proveedorcve = proveedor.proveedorcve and
       :new.empresacve = proveedor.empresacve */
    :new.proveedorcve = proveedor.proveedorcve and
    :new.empresacve = proveedor.empresacve;
if (
  /*  */

  numrows = 0
)
then
  raise_application_error(
    -20007,
    'Cannot UPDATE compra because proveedor does not exist.'
  );
end if;


/* empleado R/21 compra ON CHILD UPDATE RESTRICT */
select count(*) into numrows
  from empleado
  where
    /* :new.empleadocve = empleado.empleadocve and
       :new.departamentocve = empleado.departamentocve and
       :new.cargocve = empleado.cargocve */
    :new.empleadocve = empleado.empleadocve and
    :new.departamentocve = empleado.departamentocve and
    :new.cargocve = empleado.cargocve;
if (
  /*  */

  numrows = 0
)
then
  raise_application_error(
    -20007,
```

```
    'Cannot UPDATE compra because empleado does not exist.'
  );
 end if;



end;
/

create trigger tI_comprapago after INSERT on comprapago for each row

-- INSERT trigger on comprapago
declare numrows INTEGER;
begin

  /* compra R/31 comprapago ON CHILD INSERT RESTRICT */
  select count(*) into numrows
   from compra
   where
    /* :new.compracve = compra.compracve and
      :new.empleadocve = compra.empleadocve and
      :new.departamentocve = compra.departamentocve and
      :new.cargocve = compra.cargocve and
      :new.proveedorcve = compra.proveedorcve and
      :new.empresacve = compra.empresacve */
    :new.compracve = compra.compracve and
    :new.empleadocve = compra.empleadocve and
    :new.departamentocve = compra.departamentocve and
    :new.cargocve = compra.cargocve and
    :new.proveedorcve = compra.proveedorcve and
    :new.empresacve = compra.empresacve;
  if (
   /* */

   numrows = 0
  )
  then
   raise_application_error(
    -20002,
    'Cannot INSERT comprapago because compra does not exist.'
   );
  end if;


  /* cuentabanco R/30 comprapago ON CHILD INSERT RESTRICT */
  select count(*) into numrows
   from cuentabanco
   where
    /* :new.cuentabancocve = cuentabanco.cuentabancocve */
    :new.cuentabancocve = cuentabanco.cuentabancocve;
  if (
   /* */

   numrows = 0
  )
  then
   raise_application_error(
    -20002,
    'Cannot INSERT comprapago because cuentabanco does not exist.'
   );
  end if;
```

```
  /* tipopago R/29 comprapago ON CHILD INSERT RESTRICT */
  select count(*) into numrows
    from tipopago
    where
      /* :new.tipopagocve = tipopago.tipopagocve */
      :new.tipopagocve = tipopago.tipopagocve;
  if (
    /* */

    numrows = 0
  )
  then
    raise_application_error(
      -20002,
      'Cannot INSERT comprapago because tipopago does not exist.'
    );
  end if;



end;
/

create trigger tU_comprapago after UPDATE on comprapago for each row

-- UPDATE trigger on comprapago
declare numrows INTEGER;
begin

  /* compra R/31 comprapago ON CHILD UPDATE RESTRICT */
  select count(*) into numrows
    from compra
    where
      /* :new.compracve = compra.compracve and
         :new.empleadocve = compra.empleadocve and
         :new.departamentocve = compra.departamentocve and
         :new.cargocve = compra.cargocve and
         :new.proveedorcve = compra.proveedorcve and
         :new.empresacve = compra.empresacve */
      :new.compracve = compra.compracve and
      :new.empleadocve = compra.empleadocve and
      :new.departamentocve = compra.departamentocve and
      :new.cargocve = compra.cargocve and
      :new.proveedorcve = compra.proveedorcve and
      :new.empresacve = compra.empresacve;
  if (
    /* */

    numrows = 0
  )
  then
    raise_application_error(
      -20007,
      'Cannot UPDATE comprapago because compra does not exist.'
    );
  end if;


  /* cuentabanco R/30 comprapago ON CHILD UPDATE RESTRICT */
  select count(*) into numrows
    from cuentabanco
```

```
  where
    /* :new.cuentabancocve = cuentabanco.cuentabancocve */
    :new.cuentabancocve = cuentabanco.cuentabancocve;
 if (
  /*  */

  numrows = 0
 )
 then
  raise_application_error(
    -20007,
    'Cannot UPDATE comprapago because cuentabanco does not exist.'
  );
 end if;


 /* tipopago R/29 comprapago ON CHILD UPDATE RESTRICT */
 select count(*) into numrows
  from tipopago
  where
    /* :new.tipopagocve = tipopago.tipopagocve */
    :new.tipopagocve = tipopago.tipopagocve;
 if (
  /*  */

  numrows = 0
 )
 then
  raise_application_error(
    -20007,
    'Cannot UPDATE comprapago because tipopago does not exist.'
  );
 end if;



end;
/

create trigger tD_cuentabanco after DELETE on cuentabanco for each row

-- DELETE trigger on cuentabanco
declare numrows INTEGER;
begin

   /* cuentabanco R/59 facturadistribucionpago ON PARENT DELETE RESTRICT */
   select count(*) into numrows
     from facturadistribucionpago
     where
       /*  facturadistribucionpago.cuentabancocve = :old.cuentabancocve */
       facturadistribucionpago.cuentabancocve = :old.cuentabancocve;
   if (numrows > 0)
   then
     raise_application_error(
       -20001,
       'Cannot DELETE cuentabanco because facturadistribucionpago exists.'
     );
   end if;


   /* cuentabanco R/37 facturapublicacionpago ON PARENT DELETE RESTRICT */
   select count(*) into numrows
```

```
    from facturapublicacionpago
    where
      /*  facturapublicacionpago.cuentabancocve = :old.cuentabancocve */
      facturapublicacionpago.cuentabancocve = :old.cuentabancocve;
  if (numrows > 0)
  then
    raise_application_error(
      -20001,
      'Cannot DELETE cuentabanco because facturapublicacionpago exists.'
    );
  end if;


  /* cuentabanco R/30 comprapago ON PARENT DELETE RESTRICT */
  select count(*) into numrows
    from comprapago
    where
      /*  comprapago.cuentabancocve = :old.cuentabancocve */
      comprapago.cuentabancocve = :old.cuentabancocve;
  if (numrows > 0)
  then
    raise_application_error(
      -20001,
      'Cannot DELETE cuentabanco because comprapago exists.'
    );
  end if;



end;
/

create trigger tU_cuentabanco after UPDATE on cuentabanco for each row

-- UPDATE trigger on cuentabanco
declare numrows INTEGER;
begin

  /* cuentabanco R/59 facturadistribucionpago ON PARENT UPDATE RESTRICT */
  if
    /* :old.cuentabancocve <> :new.cuentabancocve */
    :old.cuentabancocve <> :new.cuentabancocve
  then
    select count(*) into numrows
      from facturadistribucionpago
      where
        /*  facturadistribucionpago.cuentabancocve = :old.cuentabancocve */
        facturadistribucionpago.cuentabancocve = :old.cuentabancocve;
    if (numrows > 0)
    then
      raise_application_error(
        -20005,
        'Cannot UPDATE cuentabanco because facturadistribucionpago exists.'
      );
    end if;
  end if;


  /* cuentabanco R/37 facturapublicacionpago ON PARENT UPDATE RESTRICT */
  if
    /* :old.cuentabancocve <>  new.cuentabancocve */
    :old.cuentabancocve <> :new.cuentabancocve
```

```
then
 select count(*) into numrows
  from facturapublicacionpago
  where
   /*  facturapublicacionpago.cuentabancocve = :old.cuentabancocve */
   facturapublicacionpago.cuentabancocve = :old.cuentabancocve;
 if (numrows > 0)
 then
  raise_application_error(
   -20005,
   'Cannot UPDATE cuentabanco because facturapublicacionpago exists.'
  );
 end if;
end if;


/* cuentabanco R/30 comprapago ON PARENT UPDATE RESTRICT */
if
 /* :old.cuentabancocve <> :new.cuentabancocve */
 :old.cuentabancocve <> :new.cuentabancocve
then
 select count(*) into numrows
  from comprapago
  where
   /*  comprapago.cuentabancocve = :old.cuentabancocve */
   comprapago.cuentabancocve = :old.cuentabancocve;
 if (numrows > 0)
 then
  raise_application_error(
   -20005,
   'Cannot UPDATE cuentabanco because comprapago exists.'
  );
 end if;
end if;


end;
/

create trigger tD_departamento after DELETE on departamento for each row

-- DELETE trigger on departamento
declare numrows INTEGER;
begin

  /* departamento R/5 empleado ON PARENT DELETE RESTRICT */
  select count(*) into numrows
   from empleado
   where
    /*  empleado.departamentocve = :old.departamentocve */
    empleado.departamentocve = :old.departamentocve;
  if (numrows > 0)
  then
   raise_application_error(
    -20001,
    'Cannot DELETE departamento because empleado exists.'
   );
  end if;
```

```
end;
/

create trigger tU_departamento after UPDATE on departamento for each row

-- UPDATE trigger on departamento
declare numrows INTEGER;
begin

 /* departamento R/5 empleado ON PARENT UPDATE RESTRICT */
 if
  /* :old.departamentocve <> :new.departamentocve */
  :old.departamentocve <> :new.departamentocve
 then
  select count(*) into numrows
   from empleado
   where
    /* empleado.departamentocve = :old.departamentocve */
    empleado.departamentocve = :old.departamentocve;
  if (numrows > 0)
  then
   raise_application_error(
    -20005,
    'Cannot UPDATE departamento because empleado exists.'
   );
  end if;
 end if;



end;
/

create trigger tD_descuento after DELETE on descuento for each row

-- DELETE trigger on descuento
declare numrows INTEGER;
begin

  /* descuento R/34 facturapublicacion ON PARENT DELETE RESTRICT */
  select count(*) into numrows
   from facturapublicacion
   where
    /* facturapublicacion.descuentocve = :old.descuentocve and
       facturapublicacion.clientedifesacve = :old.clientedifesacve and
       facturapublicacion.tipoclientedifesacve = :old.tipoclientedifesacve and
       facturapublicacion.empresacve = :old.empresacve and
       facturapublicacion.situacionclientecve = :old.situacionclientecve and
       facturapublicacion.tipoclientecve = :old.tipoclientecve and
       facturapublicacion.publicacioncve = :old.publicacioncve and
       facturapublicacion.proveedorcve = :old.proveedorcve */
    facturapublicacion.descuentocve = :old.descuentocve and
    facturapublicacion.clientedifesacve = :old.clientedifesacve and
    facturapublicacion.tipoclientedifesacve = :old.tipoclientedifesacve and
    facturapublicacion.empresacve = :old.empresacve and
    facturapublicacion.situacionclientecve = :old.situacionclientecve and
    facturapublicacion.tipoclientecve = :old.tipoclientecve and
    facturapublicacion.publicacioncve = :old.publicacioncve and
    facturapublicacion.proveedorcve = :old.proveedorcve;
  if (numrows > 0)
  then
   raise_application_error(
```

```
    -20001,
    'Cannot DELETE descuento because facturapublicacion exists.'
   );
  end if;



end;
/

create trigger tI_descuento after INSERT on descuento for each row

-- INSERT trigger on descuento
declare numrows INTEGER;
begin

  /* publicacion R/20 descuento ON CHILD INSERT RESTRICT */
  select count(*) into numrows
   from publicacion
   where
    /* :new.publicacioncve = publicacion.publicacioncve and
      :new.proveedorcve = publicacion.proveedorcve and
      :new.empresacve = publicacion.empresacve */
    :new.publicacioncve = publicacion.publicacioncve and
    :new.proveedorcve = publicacion.proveedorcve and
    :new.empresacve = publicacion.empresacve;
  if (
   /* */

   numrows = 0
  )
  then
   raise_application_error(
    -20002,
    'Cannot INSERT descuento because publicacion does not exist.'
   );
  end if;


  /* clientedifesa R/19 descuento ON CHILD INSERT RESTRICT */
  select count(*) into numrows
   from clientedifesa
   where
    /* :new.clientedifesacve = clientedifesa.clientedifesacve and
      :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
      :new.empresacve = clientedifesa.empresacve and
      :new.situacionclientecve = clientedifesa.situacionclientecve and
      :new.tipoclientecve = clientedifesa.tipoclientecve */
    :new.clientedifesacve = clientedifesa.clientedifesacve and
    :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
    :new.empresacve = clientedifesa.empresacve and
    :new.situacionclientecve = clientedifesa.situacionclientecve and
    :new.tipoclientecve = clientedifesa.tipoclientecve;
  if (
   /* */

   numrows = 0
  )
  then
   raise_application_error(
    -20002,
    'Cannot INSERT descuento because clientedifesa does not exist.'
```

```
    );
   end if;


end;
/

create trigger tU_descuento after UPDATE on descuento for each row

-- UPDATE trigger on descuento
declare numrows INTEGER;
begin

 /* descuento R/34 facturapublicacion ON PARENT UPDATE RESTRICT */
 if
  /* :old.descuentocve <> :new.descuentocve or
     :old.clientedifesacve <> :new.clientedifesacve or
     :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
     :old.empresacve <> :new.empresacve or
     :old.situacionclientecve <> :new.situacionclientecve or
     :old.tipoclientecve <> :new.tipoclientecve or
     :old.publicacioncve <> :new.publicacioncve or
     :old.proveedorcve <> :new.proveedorcve */
  :old.descuentocve <> :new.descuentocve or
  :old.clientedifesacve <> :new.clientedifesacve or
  :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
  :old.empresacve <> :new.empresacve or
  :old.situacionclientecve <> :new.situacionclientecve or
  :old.tipoclientecve <> :new.tipoclientecve or
  :old.publicacioncve <> :new.publicacioncve or
  :old.proveedorcve <> :new.proveedorcve
 then
  select count(*) into numrows
   from facturapublicacion
   where
    /* facturapublicacion.descuentocve = :old.descuentocve and
       facturapublicacion.clientedifesacve = :old.clientedifesacve and
       facturapublicacion.tipoclientedifesacve = :old.tipoclientedifesacve and
       facturapublicacion.empresacve = :old.empresacve and
       facturapublicacion.situacionclientecve = :old.situacionclientecve and
       facturapublicacion.tipoclientecve = :old.tipoclientecve and
       facturapublicacion.publicacioncve = :old.publicacioncve and
       facturapublicacion.proveedorcve = :old.proveedorcve */
     facturapublicacion.descuentocve = :old.descuentocve and
     facturapublicacion.clientedifesacve = :old.clientedifesacve and
     facturapublicacion.tipoclientedifesacve = :old.tipoclientedifesacve and
     facturapublicacion.empresacve = :old.empresacve and
     facturapublicacion.situacionclientecve = :old.situacionclientecve and
     facturapublicacion.tipoclientecve = :old.tipoclientecve and
     facturapublicacion.publicacioncve = :old.publicacioncve and
     facturapublicacion.proveedorcve = :old.proveedorcve;
  if (numrows > 0)
  then
   raise_application_error(
    -20005,
    'Cannot UPDATE descuento because facturapublicacion exists.'
   );
  end if;
 end if;
```

```
/* publicacion R/20 descuento ON CHILD UPDATE RESTRICT */
select count(*) into numrows
  from publicacion
  where
    /* :new.publicacioncve = publicacion.publicacioncve and
       :new.proveedorcve = publicacion.proveedorcve and
       :new.empresacve = publicacion.empresacve */
    :new.publicacioncve = publicacion.publicacioncve and
    :new.proveedorcve = publicacion.proveedorcve and
    :new.empresacve = publicacion.empresacve;
if (
  /*  */

  numrows = 0
)
then
  raise_application_error(
    -20007,
    'Cannot UPDATE descuento because publicacion does not exist.'
  );
end if;


/* clientedifesa R/19 descuento ON CHILD UPDATE RESTRICT */
select count(*) into numrows
  from clientedifesa
  where
    /* :new.clientedifesacve = clientedifesa.clientedifesacve and
       :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
       :new.empresacve = clientedifesa.empresacve and
       :new.situacionclientecve = clientedifesa.situacionclientecve and
       :new.tipoclientecve = clientedifesa.tipoclientecve */
    :new.clientedifesacve = clientedifesa.clientedifesacve and
    :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
    :new.empresacve = clientedifesa.empresacve and
    :new.situacionclientecve = clientedifesa.situacionclientecve and
    :new.tipoclientecve = clientedifesa.tipoclientecve;
if (
  /*  */

  numrows = 0
)
then
  raise_application_error(
    -20007,
    'Cannot UPDATE descuento because clientedifesa does not exist.'
  );
end if;



end;
/

create trigger tI_devolucion after INSERT on devolucion for each row

-- INSERT trigger on devolucion
declare numrows INTEGER;
begin

   /* clientedifesa R/39 devolucion ON CHILD INSERT RESTRICT */
   select count(*) into numrows
```

```
  from clientedifesa
  where
   /* :new.clientedifesacve = clientedifesa.clientedifesacve and
      :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
      :new.empresacve = clientedifesa.empresacve and
      :new.situacionclientecve = clientedifesa.situacionclientecve and
      :new.tipoclientecve = clientedifesa.tipoclienteeve */
    :new.clientedifesacve = clientedifesa.clientedifesacve and
    :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
    :new.empresacve = clientedifesa.empresacve and
    :new.situacionclientecve = clientedifesa.situacionclientecve and
    :new.tipoclientecve = clientedifesa.tipoclientecve;
 if (
  /* */

  numrows = 0
 )
 then
  raise_application_error(
   -20002,
   'Cannot INSERT devolucion because clientedifesa does not exist.'
  );
 end if;


 /* edicion R/38 devolucion ON CHILD INSERT RESTRICT */
 select count(*) into numrows
  from edicion
  where
   /* :new.edicioncve = edicion.edicioncve and
      :new.publicacioncve = edicion.publicacioncve and
      :new.proveedorcve = edicion.proveedorcve and
      :new.empresacve = edicion.empresacve */
    :new.edicioncve = edicion.edicioncve and
    :new.publicacioncve = edicion.publicacioncve and
    :new.proveedorcve = edicion.proveedorcve and
    :new.empresacve = edicion.empresacve;
 if (
  /* */

  numrows = 0
 )
 then
  raise_application_error(
   -20002,
   'Cannot INSERT devolucion because edicion does not exist.'
  );
 end if;



end;
/

create trigger tU_devolucion after UPDATE on devolucion for each row

-- UPDATE trigger on devolucion
declare numrows INTEGER;
begin

 /* clientedifesa R/39 devolucion ON CHILD UPDATE RESTRICT */
 select count(*) into numrows
```

```
  from clientedifesa
  where
    /* :new.clientedifesacve = clientedifesa.clientedifesacve and
       :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
       :new.empresacve = clientedifesa.empresacve and
       :new.situacionclientecve = clientedifesa.situacionclientecve and
       :new.tipoclientecve = clientedifesa.tipoclientecve */
     :new.clientedifesacve = clientedifesa.clientedifesacve and
     :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
     :new.empresacve = clientedifesa.empresacve and
     :new.situacionclientecve = clientedifesa.situacionclientecve and
     :new.tipoclientecve = clientedifesa.tipoclientecve;
 if (
  /*  */

  numrows = 0
 )
 then
  raise_application_error(
    -20007,
    'Cannot UPDATE devolucion because clientedifesa does not exist.'
  );
 end if;


 /* edicion R/38 devolucion ON CHILD UPDATE RESTRICT */
 select count(*) into numrows
  from edicion
  where
    /* :new.edicioncve = edicion.edicioncve and
       :new.publicacioncve = edicion.publicacioncve and
       :new.proveedorcve = edicion.proveedorcve and
       :new.empresacve = edicion.empresacve */
     :new.edicioncve = edicion.edicioncve and
     :new.publicacioncve = edicion.publicacioncve and
     :new.proveedorcve = edicion.proveedorcve and
     :new.empresacve = edicion.empresacve;
 if (
  /*  */

  numrows = 0
 )
 then
  raise_application_error(
    -20007,
    'Cannot UPDATE devolucion because edicion does not exist.'
  );
 end if;



end;
/

create trigger tD_distribucioncosto after DELETE on distribucioncosto for each row

-- DELETE trigger on distribucioncosto
declare numrows INTEGER;
begin

    /* distribucioncosto R/55 facturadistribucion ON PARENT DELETE RESTRICT */
    select count(*) into numrows
```

```
  from facturadistribucion
  where
   /*  facturadistribucion.distribucioncostocve = :old.distribucioncostocve and
      facturadistribucion.clientedifesacve = :old.clientedifesacve and
      facturadistribucion.tipoclientedifesacve = :old.tipoclientedifesacve and
      facturadistribucion.situacionclientecve = :old.situacionclientecve and
      facturadistribucion.empresacve = :old.empresacve and
      facturadistribucion.tipoclientecve = :old.tipoclientecve and
      facturadistribucion.tipoembarquecve = :old.tipoembarquecve */
    facturadistribucion.distribucioncostocve = :old.distribucioncostocve and
    facturadistribucion.clientedifesacve = :old.clientedifesacve and
    facturadistribucion.tipoclientedifesacve = :old.tipoclientedifesacve and
    facturadistribucion.situacionclientecve = :old.situacionclientecve and
    facturadistribucion.empresacve = :old.empresacve and
    facturadistribucion.tipoclientecve = :old.tipoclientecve and
    facturadistribucion.tipoembarquecve = :old.tipoembarquecve;
  if (numrows > 0)
  then
   raise_application_error(
    -20001,
    'Cannot DELETE distribucioncosto because facturadistribucion exists.'
   );
  end if;



end;
/

create trigger tI_distribucioncosto after INSERT on distribucioncosto for each row

-- INSERT trigger on distribucioncosto
declare numrows INTEGER;
begin

  /* tipoembarque R/54 distribucioncosto ON CHILD INSERT RESTRICT */
  select count(*) into numrows
   from tipoembarque
   where
    /* :new.tipoembarquecve = tipoembarque.tipoembarquecve */
    :new.tipoembarquecve = tipoembarque.tipoembarquecve;
  if (
   /*  */

   numrows = 0
  )
  then
   raise_application_error(
    -20002,
    'Cannot INSERT distribucioncosto because tipoembarque does not exist.'
   );
  end if;


  /* clientedifesa R/53 distribucioncosto ON CHILD INSERT RESTRICT */
  select count(*) into numrows
   from clientedifesa
   where
    /* :new.clientedifesacve = clientedifesa.clientedifesacve and
      :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
      :new.empresacve = clientedifesa.empresacve and
      :new.situacionclientecve = clientedifesa.situacionclientecve and
```

```
      :new.tipoclientecve = clientedifesa.tipoclientecve */
    :new.clientedifesacve = clientedifesa.clientedifesacve and
    :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
    :new.empresacve = clientedifesa.empresacve and
    :new.situacionclientecve = clientedifesa.situacionclientecve and
    :new.tipoclientecve = clientedifesa.tipoclientecve;
  if (
   /* */

    numrows = 0
  )
  then
   raise_application_error(
    -20002,
    'Cannot INSERT distribucioncosto because clientedifesa does not exist.'
   );
  end if;



end;
/

create trigger tU_distribucioncosto after UPDATE on distribucioncosto for each row

-- UPDATE trigger on distribucioncosto
declare numrows INTEGER;
begin

 /* distribucioncosto R/55 facturadistribucion ON PARENT UPDATE RESTRICT */
 if
  /* :old.distribucioncostocve <> :new.distribucioncostocve or
    :old.clientedifesacve <> :new.clientedifesacve or
    :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
    :old.situacionclientecve <> :new.situacionclientecve or
    :old.empresacve <> :new.empresacve or
    :old.tipoclientecve <> :new.tipoclientecve or
    :old.tipoembarquecve <> :new.tipoembarquecve */
  :old.distribucioncostocve <> :new.distribucioncostocve or
  :old.clientedifesacve <> :new.clientedifesacve or
  :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
  :old.situacionclientecve <> :new.situacionclientecve or
  :old.empresacve <> :new.empresacve or
  :old.tipoclientecve <> :new.tipoclientecve or
  :old.tipoembarquecve <> :new.tipoembarquecve
 then
  select count(*) into numrows
   from facturadistribucion
   where
    /* facturadistribucion.distribucioncostocve = :old.distribucioncostocve and
       facturadistribucion.clientedifesacve = :old.clientedifesacve and
       facturadistribucion.tipoclientedifesacve = :old.tipoclientedifesacve and
       facturadistribucion.situacionclientecve = :old.situacionclientecve and
       facturadistribucion.empresacve = :old.empresacve and
       facturadistribucion.tipoclientecve = :old.tipoclientecve and
       facturadistribucion.tipoembarquecve = :old.tipoembarquecve */
     facturadistribucion.distribucioncostocve = :old.distribucioncostocve and
     facturadistribucion.clientedifesacve = :old.clientedifesacve and
     facturadistribucion.tipoclientedifesacve = :old.tipoclientedifesacve and
     facturadistribucion.situacionclientecve = :old.situacionclientecve and
     facturadistribucion.empresacve = :old.empresacve and
     facturadistribucion.tipoclientecve = :old.tipoclientecve and
```

```
    facturadistribucion.tipoembarquecve = :old.tipoembarquecve;
 if (numrows > 0)
 then
  raise_application_error(
   -20005,
   'Cannot UPDATE distribucioncosto because facturadistribucion exists.'
  );
 end if;
end if;


/* tipoembarque R/54 distribucioncosto ON CHILD UPDATE RESTRICT */
select count(*) into numrows
 from tipoembarque
 where
  /* :new.tipoembarquecve = tipoembarque.tipoembarquecve */
  :new.tipoembarquecve = tipoembarque.tipoembarquecve;
if (
 /* */

 numrows = 0
)
then
 raise_application_error(
  -20007,
  'Cannot UPDATE distribucioncosto because tipoembarque does not exist.'
 );
end if;


/* clientedifesa R/53 distribucioncosto ON CHILD UPDATE RESTRICT */
select count(*) into numrows
 from clientedifesa
 where
  /* :new.clientedifesacve = clientedifesa.clientedifesacve and
     :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
     :new.empresacve = clientedifesa.empresacve and
     :new.situacionclientecve = clientedifesa.situacionclientecve and
     :new.tipoclientecve = clientedifesa.tipoclientecve */
  :new.clientedifesacve = clientedifesa.clientedifesacve and
  :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
  :new.empresacve = clientedifesa.empresacve and
  :new.situacionclientecve = clientedifesa.situacionclientecve and
  :new.tipoclientecve = clientedifesa.tipoclientecve;
if (
 /* */

 numrows = 0
)
then
 raise_application_error(
  -20007,
  'Cannot UPDATE distribucioncosto because clientedifesa does not exist.'
 );
end if;



end;
/

create trigger tD_edicion after DELETE on edicion for each row
```

```
-- DELETE trigger on edicion
declare numrows INTEGER;
begin

   /* edicion R/38 devolucion ON PARENT DELETE RESTRICT */
   select count(*) into numrows
    from devolucion
    where
     /*  devolucion.edicioncve = :old.edicioncve and
        devolucion.publicacioncve = :old.publicacioncve and
        devolucion.proveedorcve = :old.proveedorcve and
        devolucion.empresacve = :old.empresacve */
     devolucion.edicioncve = :old.edicioncve and
     devolucion.publicacioncve = :old.publicacioncve and
     devolucion.proveedorcve = :old.proveedorcve and
     devolucion.empresacve = :old.empresacve;
   if (numrows > 0)
   then
    raise_application_error(
     -20001,
     'Cannot DELETE edicion because devolucion exists.'
    );
   end if;


   /* edicion R/27 almacenentradapublicacion ON PARENT DELETE RESTRICT */
   select count(*) into numrows
    from almacenentradapublicacion
    where
     /*  almacenentradapublicacion.edicioncve = :old.edicioncve and
        almacenentradapublicacion.publicacioncve = :old.publicacioncve and
        almacenentradapublicacion.proveedorcve = :old.proveedorcve and
        almacenentradapublicacion.empresacve = :old.empresacve */
     almacenentradapublicacion.edicioncve = :old.edicioncve and
     almacenentradapublicacion.publicacioncve = :old.publicacioncve and
     almacenentradapublicacion.proveedorcve = :old.proveedorcve and
     almacenentradapublicacion.empresacve = :old.empresacve;
   if (numrows > 0)
   then
    raise_application_error(
     -20001,
     'Cannot DELETE edicion because almacenentradapublicacion exists.'
    );
   end if;



end;
/

create trigger tI_edicion after INSERT on edicion for each row

-- INSERT trigger on edicion
declare numrows INTEGER;
begin

   /* publicacion R/10 edicion ON CHILD INSERT RESTRICT */
   select count(*) into numrows
    from publicacion
    where
     /* :new.publicacioncve = publicacion.publicacioncve and
```

```
      :new.proveedorcve = publicacion.proveedorcve and
      :new.empresacve = publicacion.empresacve */
    :new.publicacioncve = publicacion.publicacioncve and
    :new.proveedorcve = publicacion.proveedorcve and
    :new.empresacve = publicacion.empresacve;
 if (
  /* */

  numrows = 0
 )
 then
  raise_application_error(
   -20002,
   'Cannot INSERT edicion because publicacion does not exist.'
  );
 end if;



end;
/

create trigger tU_edicion after UPDATE on edicion for each row

-- UPDATE trigger on edicion
declare numrows INTEGER;
begin

 /* edicion R/38 devolucion ON PARENT UPDATE RESTRICT */
 if
  /* :old.edicioncve <> :new.edicioncve or
    :old.publicacioncve <> :new.publicacioncve or
    :old.proveedorcve <> :new.proveedorcve or
    :old.empresacve <> :new.empresacve */
  :old.edicioncve <> :new.edicioncve or
  :old.publicacioncve <> :new.publicacioncve or
  :old.proveedorcve <> :new.proveedorcve or
  :old.empresacve <> :new.empresacve
 then
  select count(*) into numrows
   from devolucion
   where
    /* devolucion.edicioncve = :old.edicioncve and
       devolucion.publicacioncve = :old.publicacioncve and
       devolucion.proveedorcve = :old.proveedorcve and
       devolucion.empresacve = :old.empresacve */
    devolucion.edicioncve = :old.edicioncve and
    devolucion.publicacioncve = :old.publicacioncve and
    devolucion.proveedorcve = :old.proveedorcve and
    devolucion.empresacve = :old.empresacve;
  if (numrows > 0)
  then
   raise_application_error(
    -20005,
    'Cannot UPDATE edicion because devolucion exists.'
   );
  end if;
 end if;


 /* edicion R/27 almacenentradapublicacion ON PARENT UPDATE RESTRICT */
 if
```

```
  /* :old.edicioncve <> :new.edicioncve or
     :old.publicacioncve <> :new.publicacioncve or
     :old.proveedorcve <> :new.proveedorcve or
     :old.empresacve <> :new.empresacve */
   :old.edicioncve <> :new.edicioncve or
   :old.publicacioncve <> :new.publicacioncve or
   :old.proveedorcve <> :new.proveedorcve or
   :old.empresacve <> :new.empresacve
 then
   select count(*) into numrows
     from almacenentradapublicacion
     where
       /*  almacenentradapublicacion.edicioncve = :old.edicioncve and
           almacenentradapublicacion.publicacioncve = :old.publicacioncve and
           almacenentradapublicacion.proveedorcve = :old.proveedorcve and
           almacenentradapublicacion.empresacve = :old.empresacve */
       almacenentradapublicacion.edicioncve = :old.edicioncve and
       almacenentradapublicacion.publicacioncve = :old.publicacioncve and
       almacenentradapublicacion.proveedorcve = :old.proveedorcve and
       almacenentradapublicacion.empresacve = :old.empresacve;
   if (numrows > 0)
   then
     raise_application_error(
       -20005,
       'Cannot UPDATE edicion because almacenentradapublicacion exists.'
     );
   end if;
 end if;


 /* publicacion R/10 edicion ON CHILD UPDATE RESTRICT */
 select count(*) into numrows
   from publicacion
   where
     /* :new.publicacioncve = publicacion.publicacioncve and
        :new.proveedorcve = publicacion.proveedorcve and
        :new.empresacve = publicacion.empresacve */
     :new.publicacioncve = publicacion.publicacioncve and
     :new.proveedorcve = publicacion.proveedorcve and
     :new.empresacve = publicacion.empresacve;
 if (
   /*  */

   numrows = 0
 )
 then
   raise_application_error(
     -20007,
     'Cannot UPDATE edicion because publicacion does not exist.'
   );
 end if;



end;
/

create trigger tD_empleado after DELETE on empleado for each row

-- DELETE trigger on empleado
declare numrows INTEGER;
begin
```

```
/* empleado R/42 ruta ON PARENT DELETE RESTRICT */
select count(*) into numrows
  from ruta
  where
    /*  ruta.empleadocve = :old.empleadocve and
        ruta.departamentocve = :old.departamentocve and
        ruta.cargocve = :old.cargocve */
    ruta.empleadocve = :old.empleadocve and
    ruta.departamentocve = :old.departamentocve and
    ruta.cargocve = :old.cargocve;
if (numrows > 0)
then
  raise_application_error(
    -20001,
    'Cannot DELETE empleado because ruta exists.'
  );
end if;


/* empleado R/21 compra ON PARENT DELETE RESTRICT */
select count(*) into numrows
  from compra
  where
    /*  compra.empleadocve = :old.empleadocve and
        compra.departamentocve = :old.departamentocve and
        compra.cargocve = :old.cargocve */
    compra.empleadocve = :old.empleadocve and
    compra.departamentocve = :old.departamentocve and
    compra.cargocve = :old.cargocve;
if (numrows > 0)
then
  raise_application_error(
    -20001,
    'Cannot DELETE empleado because compra exists.'
  );
end if;


/* empleado R/17 usuario ON PARENT DELETE RESTRICT */
select count(*) into numrows
  from usuario
  where
    /*  usuario.empleadocve = :old.empleadocve and
        usuario.departamentocve = :old.departamentocve and
        usuario.cargocve = :old.cargocve */
    usuario.empleadocve = :old.empleadocve and
    usuario.departamentocve = :old.departamentocve and
    usuario.cargocve = :old.cargocve;
if (numrows > 0)
then
  raise_application_error(
    -20001,
    'Cannot DELETE empleado because usuario exists.'
  );
end if;



end;
/
```

```
create trigger tI_empleado after INSERT on empleado for each row

-- INSERT trigger on empleado
declare numrows INTEGER;
begin

   /* cargo R/6 empleado ON CHILD INSERT RESTRICT */
   select count(*) into numrows
     from cargo
     where
       /* :new.cargocve = cargo.cargocve */
       :new.cargocve = cargo.cargocve;
   if (
     /* */

     numrows = 0
   )
   then
     raise_application_error(
       -20002,
       'Cannot INSERT empleado because cargo does not exist.'
     );
   end if;


   /* departamento R/5 empleado ON CHILD INSERT RESTRICT */
   select count(*) into numrows
     from departamento
     where
       /* :new.departamentocve = departamento.departamentocve */
       :new.departamentocve = departamento.departamentocve;
   if (
     /* */

     numrows = 0
   )
   then
     raise_application_error(
       -20002,
       'Cannot INSERT empleado because departamento does not exist.'
     );
   end if;


end;
/

create trigger tU_empleado after UPDATE on empleado for each row

-- UPDATE trigger on empleado
declare numrows INTEGER;
begin

  /* empleado R/42 ruta ON PARENT UPDATE RESTRICT */
  if
    /* :old.empleadocve <> :new.empleadocve or
       :old.departamentocve <> :new.departamentocve or
       :old.cargocve <> :new.cargocve */
    :old.empleadocve <> :new.empleadocve or
    :old.departamentocve <> :new.departamentocve or
    :old.cargocve <> :new.cargocve
```

```
then
 select count(*) into numrows
  from ruta
  where
   /*  ruta.empleadocve = :old.empleadocve and
      ruta.departamentocve = :old.departamentocve and
      ruta.cargocve = :old.cargocve */
    ruta.empleadocve = :old.empleadocve and
    ruta.departamentocve = :old.departamentocve and
    ruta.cargocve = :old.cargocve;
 if (numrows > 0)
 then
  raise_application_error(
   -20005,
   'Cannot UPDATE empleado because ruta exists.'
  );
 end if;
end if;


/* empleado R/21 compra ON PARENT UPDATE RESTRICT */
if
 /* :old.empleadocve <> :new.empleadocve or
   :old.departamentocve <> :new.departamentocve or
   :old.cargocve <> :new.cargocve */
 :old.empleadocve <> :new.empleadocve or
 :old.departamentocve <> :new.departamentocve or
 :old.cargocve <> :new.cargocve
then
 select count(*) into numrows
  from compra
  where
   /*  compra.empleadocve = :old.empleadocve and
      compra.departamentocve = :old.departamentocve and
      compra.cargocve = :old.cargocve */
    compra.empleadocve = :old.empleadocve and
    compra.departamentocve = :old.departamentocve and
    compra.cargocve = :old.cargocve;
 if (numrows > 0)
 then
  raise_application_error(
   -20005,
   'Cannot UPDATE empleado because compra exists.'
  );
 end if;
end if;


/* empleado R/17 usuario ON PARENT UPDATE RESTRICT */
if
 /* :old.empleadocve <> :new.empleadocve or
   :old.departamentocve <> :new.departamentocve or
   :old.cargocve <> :new.cargocve */
 :old.empleadocve <> :new.empleadocve or
 :old.departamentocve <> :new.departamentocve or
 :old.cargocve <> :new.cargocve
then
 seleet count(*) into numrows
  from usuario
  where
   /*  usuario.empleadocve = :old.empleadocve and
      usuario.departamentocve = :old.departamentocve and
```

```
        usuario.cargocve = :old.cargocve */
      usuario.empleadocve = :old.empleadocve and
      usuario.departamentocve = :old.departamentocve and
      usuario.cargocve = :old.cargocve;
  if (numrows > 0)
  then
   raise_application_error(
     -20005,
     'Cannot UPDATE empleado because usuario exists.'
   );
  end if;
end if;


/* cargo R/6 empleado ON CHILD UPDATE RESTRICT */
select count(*) into numrows
  from cargo
  where
    /* :new.cargocve = cargo.cargocve */
    :new.cargocve = cargo.cargocve;
if (
  /* */

  numrows = 0
)
then
  raise_application_error(
    -20007,
    'Cannot UPDATE empleado because cargo does not exist.'
  );
end if;


/* departamento R/5 empleado ON CHILD UPDATE RESTRICT */
select count(*) into numrows
  from departamento
  where
    /* :new.departamentocve = departamento.departamentocve */
    :new.departamentocve = departamento.departamentocve;
if (
  /* */

  numrows = 0
)
then
  raise_application_error(
    -20007,
    'Cannot UPDATE empleado because departamento does not exist.'
  );
end if;



end;
/

create trigger tD_empresa after DELETE on empresa for each row

-- DELETE trigger on empresa
declare numrows INTEGER;
begin
```

```
  /* empresa R/18 proveedor ON PARENT DELETE RESTRICT */
  select count(*) into numrows
    from proveedor
    where
      /*  proveedor.empresacve = :old.empresacve */
      proveedor.empresacve = :old.empresacve;
  if (numrows > 0)
  then
    raise_application_error(
      -20001,
      'Cannot DELETE empresa because proveedor exists.'
    );
  end if;


  /* empresa R/11 clientedifesa ON PARENT DELETE RESTRICT */
  select count(*) into numrows
    from clientedifesa
    where
      /*  clientedifesa.empresacve = :old.empresacve */
      clientedifesa.empresacve = :old.empresacve;
  if (numrows > 0)
  then
    raise_application_error(
      -20001,
      'Cannot DELETE empresa because clientedifesa exists.'
    );
  end if;



end;
/

create trigger tU_empresa after UPDATE on empresa for each row

-- UPDATE trigger on empresa
declare numrows INTEGER;
begin

  /* empresa R/18 proveedor ON PARENT UPDATE RESTRICT */
  if
    /* :old.empresacve <> :new.empresacve */
    :old.empresacve <> :new.empresacve
  then
    select count(*) into numrows
      from proveedor
      where
        /*  proveedor.empresacve = :old.empresacve */
        proveedor.empresacve = :old.empresacve;
    if (numrows > 0)
    then
      raise_application_error(
        -20005,
        'Cannot UPDATE empresa because proveedor exists.'
      );
    end if;
  end if;


  /* empresa R/11 clientedifesa ON PARENT UPDATE RESTRICT */
  if
```

```
  /* :old.empresacve <> :new.empresacve */
  :old.empresacve <> :new.empresacve
 then
  select count(*) into numrows
   from clientedifesa
   where
    /*  clientedifesa.empresacve = :old.empresacve */
    clientedifesa.empresacve = :old.empresacve;
  if (numrows > 0)
  then
   raise_application_error(
    -20005,
    'Cannot UPDATE empresa because clientedifesa exists.'
   );
  end if;
 end if;



end;
/

create trigger tI_entrega after INSERT on entrega for each row

-- INSERT trigger on entrega
declare numrows INTEGER;
begin

   /* facturapublicacion R/35 entrega ON CHILD INSERT RESTRICT */
   select count(*) into numrows
    from facturapublicacion
    where
     /* :new.facturapublicacionnumero = facturapublicacion.facturapublicacionnumero and
        :new.clientedifesacve = facturapublicacion.clientedifesacve and
        :new.tipoclientedifesacve = facturapublicacion.tipoclientedifesacve and
        :new.empresacve = facturapublicacion.empresacve and
        :new.situacionclientecve = facturapublicacion.situacionclientecve and
        :new.tipoclientecve = facturapublicacion.tipoclientecve and
        :new.descuentocve = facturapublicacion.descuentocve and
        :new.publicacioncve = facturapublicacion.publicacioncve and
        :new.proveedorcve = facturapublicacion.proveedorcve and
        :new.puntoentregacve = facturapublicacion.puntoentregacve and
        :new.rutacve = facturapublicacion.rutacve and
        :new.empleadocve = facturapublicacion.empleadocve and
        :new.departamentocve = facturapublicacion.departamentocve and
        :new.cargocve = facturapublicacion.cargocve */
      :new.facturapublicacionnumero = facturapublicacion.facturapublicacionnumero and
      :new.clientedifesacve = facturapublicacion.clientedifesacve and
      :new.tipoclientedifesacve = facturapublicacion.tipoclientedifesacve and
      :new.empresacve = facturapublicacion.empresacve and
      :new.situacionclientecve = facturapublicacion.situacionclientecve and
      :new.tipoclientecve = facturapublicacion.tipoclientecve and
      :new.descuentocve = facturapublicacion.descuentocve and
      :new.publicacioncve = facturapublicacion.publicacioncve and
      :new.proveedorcve = facturapublicacion.proveedorcve and
      :new.puntoentregacve = facturapublicacion.puntoentregacve and
      :new.rutacve = facturapublicacion.rutacve and
      :new.empleadocve = facturapublicacion.empleadocve and
      :new.departamentocve = facturapublicacion.departamentocve and
      :new.cargocve = facturapublicacion.cargocve;
   if (
    /* */
```

```
    numrows = 0
  )
  then
    raise_application_error(
      -20002,
      'Cannot INSERT entrega because facturapublicacion does not exist.'
    );
  end if;



end;
/

create trigger tU_entrega after UPDATE on entrega for each row

-- UPDATE trigger on entrega
declare numrows INTEGER;
begin

 /* facturapublicacion R/35 entrega ON CHILD UPDATE RESTRICT */
 select count(*) into numrows
   from facturapublicacion
   where
     /* :new.facturapublicacionnumero = facturapublicacion.facturapublicacionnumero and
        :new.clientedifesacve = facturapublicacion.clientedifesacve and
        :new.tipoclientedifesacve = facturapublicacion.tipoclientedifesacve and
        :new.empresacve = facturapublicacion.empresacve and
        :new.situacionclientecve = facturapublicacion.situacionclientecve and
        :new.tipoclientecve = facturapublicacion.tipoclientecve and
        :new.descuentocve = facturapublicacion.descuentocve and
        :new.publicacioncve = facturapublicacion.publicacioncve and
        :new.proveedorcve = facturapublicacion.proveedorcve and
        :new.puntoentregacve = facturapublicacion.puntoentregacve and
        :new.rutacve = facturapublicacion.rutacve and
        :new.empleadocve = facturapublicacion.empleadocve and
        :new.departamentocve = facturapublicacion.departamentocve and
        :new.cargocve = facturapublicacion.cargocve */
     :new.facturapublicacionnumero = facturapublicacion.facturapublicacionnumero and
     :new.clientedifesacve = facturapublicacion.clientedifesacve and
     :new.tipoclientedifesacve = facturapublicacion.tipoclientedifesacve and
     :new.empresacve = facturapublicacion.empresacve and
     :new.situacionclientecve = facturapublicacion.situacionclientecve and
     :new.tipoclientecve = facturapublicacion.tipoclientecve and
     :new.descuentocve = facturapublicacion.descuentocve and
     :new.publicacioncve = facturapublicacion.publicacioncve and
     :new.proveedorcve = facturapublicacion.proveedorcve and
     :new.puntoentregacve = facturapublicacion.puntoentregacve and
     :new.rutacve = facturapublicacion.rutacve and
     :new.empleadocve = facturapublicacion.empleadocve and
     :new.departamentocve = facturapublicacion.departamentocve and
     :new.cargocve = facturapublicacion.cargocve;
 if (
   /*  */

   numrows = 0
 )
 then
   raise_application_error(
     -20007,
     'Cannot UPDATE entrega because facturapublicacion does not exist.'
```

```
  );
 end if;



end;
/

create trigger tD_estatusdistribucion after DELETE on estatusdistribucion for each row

-- DELETE trigger on estatusdistribucion
declare numrows INTEGER;
begin

   /* estatusdistribucion R/49 almacenentradadistribucion ON PARENT DELETE RESTRICT */
   select count(*) into numrows
     from almacenentradadistribucion
     where
       /*  almacenentradadistribucion.estatusdistribucioncve = :old.estatusdistribucioncve */
       almacenentradadistribucion.estatusdistribucioncve = :old.estatusdistribucioncve;
   if (numrows > 0)
   then
     raise_application_error(
       -20001,
       'Cannot DELETE estatusdistribucion because almacenentradadistribucion exists.'
     );
   end if;



end;
/

create trigger tU_estatusdistribucion after UPDATE on estatusdistribucion for each row

-- UPDATE trigger on estatusdistribucion
declare numrows INTEGER;
begin

  /* estatusdistribucion R/49 almacenentradadistribucion ON PARENT UPDATE RESTRICT */
  if
    /* :old.estatusdistribucioncve <> :new.estatusdistribucioncve */
    :old.estatusdistribucioncve <> :new.estatusdistribucioncve
  then
    select count(*) into numrows
      from almacenentradadistribucion
      where
        /*  almacenentradadistribucion.estatusdistribucioncve = :old.estatusdistribucioncve */
        almacenentradadistribucion.estatusdistribucioncve = :old.estatusdistribucioncve;
    if (numrows > 0)
    then
      raise_application_error(
        -20005,
        'Cannot UPDATE estatusdistribucion because almacenentradadistribucion exists.'
      );
    end if;
  end if;



end;
/
```

```
create trigger tD_facturadistribucion after DELETE on facturadistribucion for each row

-- DELETE trigger on facturadistribucion
declare numrows INTEGER;
begin

   /* facturadistribucion R/57 facturadistribucionpago ON PARENT DELETE RESTRICT */
   select count(*) into numrows
     from facturadistribucionpago
     where
       /*  facturadistribucionpago.relacionembarqueclave = :old.relacionembarqueclave and
          facturadistribucionpago.facturadistribucionnumero = :old.facturadistribucionnumero and
          facturadistribucionpago.clientedifesacve = :old.clientedifesacve and
          facturadistribucionpago.tipoclientedifesacve = :old.tipoclientedifesacve and
          facturadistribucionpago.empresacve = :old.empresacve and
          facturadistribucionpago.situacionclientecve = :old.situacionclientecve and
          facturadistribucionpago.tipoclientecve = :old.tipoclientecve and
          facturadistribucionpago.tipoembarquecve = :old.tipoembarquecve and
          facturadistribucionpago.distribucioncostocve = :old.distribucioncostocve */
       facturadistribucionpago.relacionembarqueclave = :old.relacionembarqueclave and
       facturadistribucionpago.facturadistribucionnumero = :old.facturadistribucionnumero and
       facturadistribucionpago.clientedifesacve = :old.clientedifesacve and
       facturadistribucionpago.tipoclientedifesacve = :old.tipoclientedifesacve and
       facturadistribucionpago.empresacve = :old.empresacve and
       facturadistribucionpago.situacionclientecve = :old.situacionclientecve and
       facturadistribucionpago.tipoclientecve = :old.tipoclientecve and
       facturadistribucionpago.tipoembarquecve = :old.tipoembarquecve and
       facturadistribucionpago.distribucioncostocve = :old.distribucioncostocve;
   if (numrows > 0)
   then
     raise_application_error(
       -20001,
       'Cannot DELETE facturadistribucion because facturadistribucionpago exists.'
     );
   end if;



end;
/

create trigger tI_facturadistribucion after INSERT on facturadistribucion for each row

-- INSERT trigger on facturadistribucion
declare numrows INTEGER;
begin

   /* distribucioncosto R/55 facturadistribucion ON CHILD INSERT RESTRICT */
   select count(*) into numrows
     from distribucioncosto
     where
       /* :new.distribucioncostocve = distribucioncosto.distribucioncostocve and
          :new.clientedifesacve = distribucioncosto.clientedifesacve and
          :new.tipoclientedifesacve = distribucioncosto.tipoclientedifesacve and
          :new.situacionclientecve = distribucioncosto.situacionclientecve and
          :new.empresacve = distribucioncosto.empresacve and
          :new.tipoclientecve = distribucioncosto.tipoclientecve and
          :new.tipoembarquecve = distribucioncosto.tipoembarquecve */
       :new.distribucioncostocve = distribucioncosto.distribucioncostocve and
       :new.clientedifesacve = distribucioncosto.clientedifesacve and
       :new.tipoclientedifesacve = distribucioncosto.tipoclientedifesacve and
```

```
    :new.situacionclientecve = distribucioncosto.situacionclientecve and
    :new.empresacve = distribucioncosto.empresacve and
    :new.tipoclientecve = distribucioncosto.tipoclientecve and
    :new.tipoembarquecve = distribucioncosto.tipoembarquecve;
  if (
    /* */

    numrows = 0
  )
  then
    raise_application_error(
      -20002,
      'Cannot INSERT facturadistribucion because distribucioncosto does not exist.'
    );
  end if;


  /* relacionembarque R/52 facturadistribucion ON CHILD INSERT RESTRICT */
  select count(*) into numrows
    from relacionembarque
    where
      /* :new.relacionembarqueclave = relacionembarque.relacionembarqueclave and
        :new.elientedifesacve = relacionembarque.clientedifesacve and
        :new.tipoclientedifesacve = relacionembarque.tipoclientedifesacve and
        :new.empresacve = relacionembarque.empresacve and
        :new.situacionclientecve = relacionembarque.situacionclienteeve and
        :new.tipoclientecve = relacionembarque.tipoclientecve and
        :new.tipoembarquecve = relacionembarque.tipoembarquecve */
      :new.relacionembarqueclave = relacionembarque.relacionembarqueclave and
      :new.clientedifesacve = relacionembarque.clientedifesacve and
      :new.tipoclientedifesacve = relacionembarque.tipoclientedifesacve and
      :new.empresacve = relacionembarque.empresacve and
      :new.situacionclientecve = relacionembarque.situacionclientecve and
      :new.tipoclientecve = relacionembarque.tipoclientecve and
      :new.tipoembarquecve = relacionembarque.tipoembarquecve;
  if (
    /* */

    numrows = 0
  )
  then
    raise_application_error(
      -20002,
      'Cannot INSERT facturadistribucion because relacionembarque does not exist.'
    );
  end if;



end;
/

create trigger tU_facturadistribucion after UPDATE on facturadistribucion for each row

-- UPDATE trigger on facturadistribucion
declare numrows INTEGER;
begin

  /* facturadistribucion R/57 facturadistribucionpago ON PARENT UPDATE RESTRICT */
  if
    /* :old.relacionembarqueclave <> :new.relacionembarqueclave or
      :old.facturadistribucionnumero <> :new.facturadistribucionnumero or
```

```
    :old.clientedifesacve <> :new.clientedifesacve or
    :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
    :old.empresacve <> :new.empresacve or
    :old.situacionclientecve <> :new.situacionclientecve or
    :old.tipoclientecve <> :new.tipoclientecve or
    :old.tipoembarquecve <> :new.tipoembarquecve or
    :old.distribucioncostocve <> :new.distribucioncostocve */
  :old.relacionembarqueclave <> :new.relacionembarqueclave or
  :old.facturadistribucionnumero <> :new.facturadistribucionnumero or
  :old.clientedifesacve <> :new.clientedifesacve or
  :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
  :old.empresacve <> :new.empresacve or
  :old.situacionclientecve <> :new.situacionclientecve or
  :old.tipoclientecve <> :new.tipoclientecve or
  :old.tipoembarquecve <> :new.tipoembarquecve or
  :old.distribucioncostocve <> :new.distribucioncostocve
then
  select count(*) into numrows
    from facturadistribucionpago
    where
      /* facturadistribucionpago.relacionembarqueclave = :old.relacionembarqueclave and
         facturadistribucionpago.facturadistribucionnumero = :old.facturadistribucionnumero and
         facturadistribucionpago.clientedifesacve = :old.clientedifesacve and
         facturadistribucionpago.tipoclientedifesacve = :old.tipoclientedifesacve and
         facturadistribucionpago.empresacve = :old.empresacve and
         facturadistribucionpago.situacionclientecve = :old.situacionclientecve and
         facturadistribucionpago.tipoclientecve = :old.tipoclientecve and
         facturadistribucionpago.tipoembarquecve = :old.tipoembarquecve and
         facturadistribucionpago.distribucioncostocve = :old.distribucioncostocve */
      facturadistribucionpago.relacionembarqueclave = :old.relacionembarqueclave and
      facturadistribucionpago.facturadistribucionnumero = :old.facturadistribucionnumero and
      facturadistribucionpago.clientedifesacve = :old.clientedifesacve and
      facturadistribucionpago.tipoclientedifesacve = :old.tipoclientedifesacve and
      facturadistribucionpago.empresacve = :old.empresacve and
      facturadistribucionpago.situacionclientecve = :old.situacionclientecve and
      facturadistribucionpago.tipoclientecve = :old.tipoclientecve and
      facturadistribucionpago.tipoembarquecve = :old.tipoembarquecve and
      facturadistribucionpago.distribucioncostocve = :old.distribucioncostocve;
  if (numrows > 0)
  then
    raise_application_error(
      -20005,
      'Cannot UPDATE facturadistribucion because facturadistribucionpago exists.'
    );
  end if;
end if;


/* distribucioncosto R/55 facturadistribucion ON CHILD UPDATE RESTRICT */
select count(*) into numrows
  from distribucioncosto
  where
    /* :new.distribucioncostocve = distribucioncosto.distribucioncostocve and
       :new.clientedifesacve = distribucioncosto.clientedifesacve and
       :new.tipoclientedifesacve = distribucioncosto.tipoclientedifesacve and
       :new.situacionclientecve = distribucioncosto.situacionclientecve and
       :new.empresacve = distribucioncosto.empresacve and
       :new.tipoclientecve = distribucioncosto.tipoclientecve and
       :new.tipoembarquecve = distribucioncosto.tipoembarquecve */
    :new.distribucioncostocve = distribucioncosto.distribucioncostocve and
    :new.clientedifesacve = distribucioncosto.clientedifesacve and
    :new.tipoclientedifesacve = distribucioncosto.tipoclientedifesacve and
```

```
    :new.situacionclientecve = distribucioncosto.situacionclientecve and
    :new.empresacve = distribucioncosto.empresacve and
    :new.tipoclientecve = distribucioncosto.tipoclientecve and
    :new.tipoembarquecve = distribucioncosto.tipoembarquecve;
if (
  /* */

  numrows = 0
)
then
  raise_application_error(
    -20007,
    'Cannot UPDATE facturadistribucion because distribucioncosto does not exist.'
  );
end if;


/* relacionembarque R/52 facturadistribucion ON CHILD UPDATE RESTRICT */
select count(*) into numrows
  from relacionembarque
  where
    /* :new.relacionembarqueclave = relacionembarque.relacionembarqueclave and
       :new.clientedifesacve = relacionembarque.clientedifesacve and
       :new.tipoclientedifesacve = relacionembarque.tipoclientedifesacve and
       :new.empresacve = relacionembarque.empresacve and
       :new.situacionclientecve = relacionembarque.situacionclientecve and
       :new.tipoclientecve = relacionembarque.tipoclientecve and
       :new.tipoembarquecve = relacionembarque.tipoembarquecve */
    :new.relacionembarqueclave = relacionembarque.relacionembarqueclave and
    :new.clientedifesacve = relacionembarque.clientedifesacve and
    :new.tipoclientedifesacve = relacionembarque.tipoclientedifesacve and
    :new.cmpresacve = relacionembarque.empresacve and
    :new.situacionclientecve = relacionembarque.situacionclientecve and
    :new.tipoclientecve = relacionembarque.tipoclientecve and
    :new.tipoembarquecve = relacionembarque.tipoembarquecve;
if (
  /* */

  numrows = 0
)
then
  raise_application_error(
    -20007,
    'Cannot UPDATE facturadistribucion because relacionembarque does not exist.'
  );
end if;



end;
/

create trigger tI_facturadistribucionpago after INSERT on facturadistribucionpago for each row

-- INSERT trigger on facturadistribucionpago
declare numrows INTEGER;
begin

    /* cuentabanco R/59 facturadistribucionpago ON CHILD INSERT RESTRICT */
    select count(*) into numrows
      from cuentabanco
      where
```

```
   /* :new.cuentabancocve = cuentabanco.cuentabancocve */
   :new.cuentabancocve = cuentabanco.cuentabancocve;
if (
  /* */

  numrows = 0
)
then
  raise_application_error(
    -20002,
    'Cannot INSERT facturadistribucionpago because cuentabanco does not exist.'
  );
end if;


/* tipopago R/58 facturadistribucionpago ON CHILD INSERT RESTRICT */
select count(*) into numrows
  from tipopago
  where
    /* :new.tipopagocve = tipopago.tipopagocve */
    :new.tipopagocve = tipopago.tipopagocve;
if (
  /* */

  numrows = 0
)
then
  raise_application_error(
    -20002,
    'Cannot INSERT facturadistribucionpago because tipopago does not exist.'
  );
end if;


/* facturadistribucion R/57 facturadistribucionpago ON CHILD INSERT RESTRICT */
select count(*) into numrows
  from facturadistribucion
  where
    /* :new.relacionembarqueclave = facturadistribucion.relacionembarqueclave and
       :new.facturadistribucionnumero = facturadistribucion.facturadistribucionnumero and
       :new.clientedifesacve = facturadistribucion.clientedifesacve and
       :new.tipoclientedifesacve = facturadistribucion.tipoclientedifesacve and
       :new.empresacve = facturadistribucion.empresacve and
       :new.situacionclientecve = facturadistribucion.situacionclientecve and
       :new.tipoclientecve = facturadistribucion.tipoclientecve and
       :new.tipoembarquecve = facturadistribucion.tipoembarquecve and
       :new.distribucioncostocve = facturadistribucion.distribucioncostocve */
    :new.relacionembarqueclave = facturadistribucion.relacionembarqueclave and
    :new.facturadistribucionnumero = facturadistribucion.facturadistribucionnumero and
    :new.clientedifesacve = facturadistribucion.clientedifesacve and
    :new.tipoclientedifesacve = facturadistribucion.tipoclientedifesacve and
    :new.empresacve = facturadistribucion.empresacve and
    :new.situacionclientecve = facturadistribucion.situacionclientecve and
    :new.tipoclientecve = facturadistribucion.tipoclientecve and
    :new.tipoembarquecve = facturadistribucion.tipoembarquecve and
    :new.distribucioncostocve = facturadistribucion.distribucioncostocve;
if (
  /* */

  numrows = 0
)
then
```

```
    raise_application_error(
      -20002,
      'Cannot INSERT facturadistribucionpago because facturadistribucion does not exist.'
    );
   end if;



end;
/

create trigger tU_facturadistribucionpago after UPDATE on facturadistribucionpago for each row

-- UPDATE trigger on facturadistribucionpago
declare numrows INTEGER;
begin

  /* cuentabanco R/59 facturadistribucionpago ON CHILD UPDATE RESTRICT */
  select count(*) into numrows
    from cuentabanco
    where
      /* :new.cuentabancocve = cuentabanco.cuentabancocve */
      :new.cuentabancocve = cuentabanco.cuentabancocve;
  if (
    /*  */

    numrows = 0
  )
  then
    raise_application_error(
      -20007,
      'Cannot UPDATE facturadistribucionpago because cuentabanco does not exist.'
    );
  end if;


  /* tipopago R/58 facturadistribucionpago ON CHILD UPDATE RESTRICT */
  select count(*) into numrows
    from tipopago
    where
      /* :new.tipopagocve = tipopago.tipopagocve */
      :new.tipopagocve = tipopago.tipopagocve;
  if (
    /*  */

    numrows = 0
  )
  then
    raise_application_error(
      -20007,
      'Cannot UPDATE facturadistribucionpago because tipopago does not exist.'
    );
  end if;


  /* facturadistribucion R/57 facturadistribucionpago ON CHILD UPDATE RESTRICT */
  select count(*) into numrows
    from facturadistribucion
    where
      /* :new.relacionembarqueclave = facturadistribucion.relacionembarqueclave and
         :new.facturadistribucionnumero = facturadistribucion.facturadistribucionnumero and
         :new.clientedifesacve = facturadistribucion.clientedifesacve and
```

```
    :new.tipoclientedifesacve = facturadistribucion.tipoclientedifesacve and
    :new.empresacve = facturadistribucion.empresacve and
    :new.situacionclientecve = facturadistribucion.situacionclientecve and
    :new.tipoclientecve = facturadistribucion.tipoclientecve and
    :new.tipoembarquecve = facturadistribucion.tipoembarquecve and
    :new.distribucioncostocve = facturadistribucion.distribucioncostocve */
  :new.relacionembarqueclave = facturadistribucion.relacioncmbarqueclave and
  :new.facturadistribucionnumero = facturadistribucion.facturadistribucionnumero and
  :new.clientedifesacve = facturadistribucion.clientedifesacve and
  :new.tipoclientedifesacve = facturadistribucion.tipoclientedifesacve and
  :new.empresacve = facturadistribucion.empresacve and
  :new.situacionclientecve = facturadistribucion.situacionclientecve and
  :new.tipoclientecve = facturadistribucion.tipoclientecve and
  :new.tipoembarquecve = facturadistribucion.tipoembarquecve and
  :new.distribucioncostocve = facturadistribucion.distribucioncostocve;
if (
 /* */

 numrows = 0
)
then
 raise_application_error(
  -20007,
  'Cannot UPDATE facturadistribucionpago because facturadistribucion does not exist.'
 );
end if;



end;
/

create trigger tD_facturapublicacion after DELETE on facturapublicacion for each row

-- DELETE trigger on facturapublicacion
declare numrows INTEGER;
begin

  /* facturapublicacion R/35 entrega ON PARENT DELETE RESTRICT */
  select count(*) into numrows
   from entrega
   where
    /* entrega.facturapublicacionnumero = :old.facturapublicacionnumero and
       entrega.clientedifesacve = :old.clientedifesacve and
       entrega.tipoclientedifesacve = :old.tipoclientedifesacve and
       entrega.empresacve = :old.empresacve and
       entrega.situacionclientecve = :old.situacionclientecve and
       entrega.tipoclientecve = :old.tipoclientecve and
       entrega.descuentocve = :old.descuentocve and
       entrega.publicacioncve = :old.publicacioncve and
       entrega.proveedorcve = :old.proveedorcve and
       entrega.puntoentregacve = :old.puntoentregacve and
       entrega.rutacve = :old.rutacve and
       entrega.empleadocve = :old.empleadocve and
       entrega.departamentocve = :old.departamentocve and
       entrega.cargocve = :old.cargocve */
     entrega.facturapublicacionnumero = :old.facturapublicacionnumero and
     entrega.clientedifesacve = :old.clientedifesacve and
     entrega.tipoclientedifesacve = :old.tipoclientedifesacve and
     entrega.empresacve = :old.empresacve and
     entrega.situacionclientecve = :old.situacionclientecve and
     entrega.tipoclientecve = :old.tipoclientecve and
```

```
    entrega.descuentocve = :old.descuentocve and
    entrega.publicacioncve = :old.publicacioncve and
    entrega.proveedorcve = :old.proveedorcve and
    entrega.puntoentregacve = :old.puntoentregacve and
    entrega.rutacve = :old.rutacve and
    entrega.empleadocve = :old.empleadocve and
    entrega.departamentocve = :old.departamentocve and
    entrega.cargocve = :old.cargocve;
  if (numrows > 0)
  then
   raise_application_error(
    -20001,
    'Cannot DELETE facturapublicacion because entrega exists.'
   );
  end if;



end;
/

create trigger tI_facturapublicacion after INSERT on facturapublicacion for each row

-- INSERT trigger on facturapublicacion
declare numrows INTEGER;
begin

  /* puntoentrega R/41 facturapublicacion ON CHILD INSERT RESTRICT */
  select count(*) into numrows
   from puntoentrega
   where
    /* :new.puntoentregacve = puntoentrega.puntoentregacve and
      :new.rutacve = puntoentrega.rutacve and
      :new.empleadocve = puntoentrega.empleadocve and
      :new.departamentocve = puntoentrega.departamentocve and
      :new.cargocve = puntoentrega.cargocve */
    :new.puntoentregacve = puntoentrega.puntoentregacve and
    :new.rutacve = puntoentrega.rutacve and
    :new.empleadocve = puntoentrega.empleadocve and
    :new.departamentocve = puntoentrega.departamentocve and
    :new.cargocve = puntoentrega.cargocve;
  if (
   /*  */

   numrows = 0
  )
  then
   raise_application_error(
    -20002,
    'Cannot INSERT facturapublicacion because puntoentrega does not exist.'
   );
  end if;


  /* descuento R/34 facturapublicacion ON CHILD INSERT RESTRICT */
  select count(*) into numrows
   from descuento
   where
    /* :new.descuentocve = descuento.descuentocve and
      :new.clientedifesacve = descuento.clientedifesacve and
      :new.tipoclientedifesacve = descuento.tipoclientedifesacve and
      :new.empresacve = descuento.empresacve and
```

```
      :new.situacionclientecve = descuento.situacionclientecve and
      :new.tipoclientecve = descuento.tipoclientecve and
      :new.publicacioncve = descuento.publicacioncve and
      :new.proveedorcve = descuento.proveedorcve */
    :new.descuentocve = descuento.descuentocve and
    :new.clientedifesacve = descuento.clientedifesacve and
    :new.tipoclientedifesacve = descuento.tipoclientedifesacve and
    :new.empresacve = descuento.empresacve and
    :new.situacionclientecve = descuento.situacionclientecve and
    :new.tipoclientecve = descuento.tipoclientecve and
    :new.publicacioncve = descuento.publicacioncve and
    :new.proveedorcve = descuento.proveedorcve;
  if (
   /* */

   numrows = 0
  )
  then
   raise_application_error(
    -20002,
    'Cannot INSERT facturapublicacion because descuento does not exist.'
   );
  end if;


  /* clientedifesa R/33 facturapublicacion ON CHILD INSERT RESTRICT */
  select count(*) into numrows
   from clientedifesa
   where
    /* :new.clientedifesacve = clientedifesa.clientedifesacve and
      :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
      :new.empresacve = clientedifesa.empresacve and
      :new.situacionclientecve = clientedifesa.situacionclientecve and
      :new.tipoclientecve = clientedifesa.tipoclientecve */
    :new.clientedifesacve = clientedifesa.clientedifesacve and
    :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
    :new.empresacve = clientedifesa.empresacve and
    :new.situacionclientecve = clientedifesa.situacionclientecve and
    :new.tipoclientecve = clientedifesa.tipoclientecve;
  if (
   /* */

   numrows = 0
  )
  then
   raise_application_error(
    -20002,
    'Cannot INSERT facturapublicacion because clientedifesa does not exist.'
   );
  end if;



end;
/

create trigger tU_facturapublicacion after UPDATE on facturapublicacion for each row

-- UPDATE trigger on facturapublicacion
declare numrows INTEGER;
begin
```

```
/* facturapublicacion R/35 entrega ON PARENT UPDATE RESTRICT */
if
 /* :old.facturapublicacionnumero <> :new.facturapublicacionnumero or
    :old.clientedifesacve <> :new.clientedifesacve or
    :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
    :old.empresacve <> :new.empresacve or
    :old.situacionclientecve <> :new.situacionclientecve or
    :old.tipoclientecve <> :new.tipoclientecve or
    :old.descuentocve <> :new.descuentocve or
    :old.publicacioncve <> :new.publicacioncve or
    :old.proveedorcve <> :new.proveedorcve or
    :old.puntoentregacve <> :new.puntoentregacve or
    :old.rutacve <> :new.rutacve or
    :old.empleadocve <> :new.empleadocve or
    :old.departamentocve <> :new.departamentocve or
    :old.cargocve <> :new.cargocve */
  :old.facturapublicacionnumero <> :new.facturapublicacionnumero or
  :old.clientedifesacve <> :new.clientedifesacve or
  :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
  :old.empresacve <> :new.empresacve or
  :old.situacionclientecve <> :new.situacionclientecve or
  :old.tipoclientecve <> :new.tipoclientecve or
  :old.descuentocve <> :new.descuentocve or
  :old.publicacioncve <> :new.publicacioncve or
  :old.proveedorcve <> :new.proveedorcve or
  :old.puntoentregacve <> :new.puntoentregacve or
  :old.rutacve <> :new.rutacve or
  :old.empleadocve <> :new.empleadocve or
  :old.departamentocve <> :new.departamentocve or
  :old.cargocve <> :new.cargocve
then
  select count(*) into numrows
    from entrega
    where
     /* entrega.facturapublicacionnumero = :old.facturapublicacionnumero and
        entrega.clientedifesacve = :old.clientedifesacve and
        entrega.tipoclientedifesacve = :old.tipoclientedifesacve and
        entrega.empresacve = :old.empresacve and
        entrega.situacionclientecve = :old.situacionclientecve and
        entrega.tipoclientecve = :old.tipoclientecve and
        entrega.descuentocve = :old.descuentocve and
        entrega.publicacioncve = :old.publicacioncve and
        entrega.proveedorcve = :old.proveedorcve and
        entrega.puntoentregacve = :old.puntoentregacve and
        entrega.rutacve = :old.rutacve and
        entrega.empleadocve = :old.empleadocve and
        entrega.departamentocve = :old.departamentocve and
        entrega.cargocve = :old.cargocve */
      entrega.facturapublicacionnumero = :old.facturapublicacionnumero and
      entrega.clientedifesacve = :old.clientedifesacve and
      entrega.tipoclientedifesacve = :old.tipoclientedifesacve and
      entrega.empresacve = :old.empresacve and
      entrega.situacionclientecve = :old.situacionclientecve and
      entrega.tipoclientecve = :old.tipoclientecve and
      entrega.descuentocve = :old.descuentocve and
      entrega.publicacioncve = :old.publicacioncve and
      entrega.proveedorcve = :old.proveedorcve and
      entrega.puntoentregacve = :old.puntoentregacve and
      entrega.rutacve = :old.rutacve and
      entrega.empleadocve = :old.empleadocve and
      entrega.departamentocve = :old.departamentocve and
      entrega.cargocve = :old.cargocve;
```

```
  if (numrows > 0)
  then
    raise_application_error(
      -20005,
      'Cannot UPDATE facturapublicacion because entrega exists.'
    );
  end if;
end if;


/* puntoentrega R/41 facturapublicacion ON CHILD UPDATE RESTRICT */
select count(*) into numrows
  from puntoentrega
  where
    /* :new.puntoentregacve = puntoentrega.puntoentregacve and
       :new.rutacve = puntoentrega.rutacve and
       :new.empleadocve = puntoentrega.empleadocve and
       :new.departamentocve = puntoentrega.departamentocve and
       :new.cargocve = puntoentrega.cargocve */
    :new.puntoentregacve = puntoentrega.puntoentregacve and
    :new.rutacve = puntoentrega.rutacve and
    :new.empleadocve = puntoentrega.empleadocve and
    :new.departamentocve = puntoentrega.departamentocve and
    :new.cargocve = puntoentrega.cargocve;
if (
  /*  */

  numrows = 0
)
then
  raise_application_error(
    -20007,
    'Cannot UPDATE facturapublicacion because puntoentrega does not exist.'
  );
end if;


/* descuento R/34 facturapublicacion ON CHILD UPDATE RESTRICT */
select count(*) into numrows
  from descuento
  where
    /* :new.descuentocve = descuento.descuentocve and
       :new.clientedifesacve = descuento.clientedifesacve and
       :new.tipoclientedifesacve = descuento.tipoclientedifesacve and
       :new.empresacve = descuento.empresacve and
       :new.situacionclientecve = descuento.situacionclientecve and
       :new.tipoclientecve = descuento.tipoclientecve and
       :new.publicacioncve = descuento.publicacioncve and
       :new.proveedorcve = descuento.proveedorcve */
    :new.descuentocve = descuento.descuentocve and
    :new.clientedifesacve = descuento.clientedifesacve and
    :new.tipoclientedifesacve = descuento.tipoclientedifesacve and
    :new.empresacve = descuento.empresacve and
    :new.situacionclientecve = descuento.situacionclientecve and
    :new.tipoclientecve = descuento.tipoclientecve and
    :new.publicacioncve = descuento.publicacioncve and
    :new.proveedorcve = descuento.proveedorcve;
if (
  /*  */

  numrows = 0
)
```

```
then
 raise_application_error(
  -20007,
  'Cannot UPDATE facturapublicacion because descuento does not exist.'
 );
end if;


/* clientedifesa R/33 facturapublicacion ON CHILD UPDATE RESTRICT */
select count(*) into numrows
 from clientedifesa
 where
  /* :new.clientedifesacve = clientedifesa.clientedifesacve and
     :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
     :new.empresacve = clientedifesa.empresacve and
     :new.situacionclientecve = clientedifesa.situacionclientecve and
     :new.tipoclientecve = clientedifesa.tipoclientecve */
   :new.clientedifesacve = clientedifesa.clientedifesacve and
   :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
   :new.empresacve = clientedifesa.empresacve and
   :new.situacionclientecve = clientedifesa.situacionclientecve and
   :new.tipoclientecve = clientedifesa.tipoclientecve;
if (
 /*  */

 numrows = 0
)
then
 raise_application_error(
  -20007,
  'Cannot UPDATE facturapublicacion because clientedifesa does not exist.'
 );
end if;



end;
/

create trigger tl_facturapublicacionpago after INSERT on facturapublicacionpago for each row

-- INSERT trigger on facturapublicacionpago
declare numrows INTEGER;
begin

  /* cuentabanco R/37 facturapublicacionpago ON CHILD INSERT RESTRICT */
  select count(*) into numrows
   from cuentabanco
   where
    /* :new.cuentabancocve = cuentabanco.cuentabancocve */
    :new.cuentabancocve = cuentabanco.cuentabancocve;
  if (
   /*  */

   numrows = 0
  )
  then
   raise_application_error(
    -20002,
    'Cannot INSERT facturapublicacionpago because cuentabanco does not exist.'
   );
  end if;
```

```
  /* tipopago R/36 facturapublicacionpago ON CHILD INSERT RESTRICT */
  select count(*) into numrows
    from tipopago
    where
      /* :new.tipopagocve = tipopago.tipopagocve */
      :new.tipopagocve = tipopago.tipopagocve;
  if (
    /* */

    numrows = 0
  )
  then
    raise_application_error(
      -20002,
      'Cannot INSERT facturapublicacionpago because tipopago does not exist.'
    );
  end if;



end;
/

create trigger tU_facturapublicacionpago after UPDATE on facturapublicacionpago for each row

-- UPDATE trigger on facturapublicacionpago
declare numrows INTEGER;
begin

  /* cuentabanco R/37 facturapublicacionpago ON CHILD UPDATE RESTRICT */
  select count(*) into numrows
    from cuentabanco
    where
      /* :new.cuentabancocve = cuentabanco.cuentabancocve */
      :new.cuentabancocve = cuentabanco.cuentabancocve;
  if (
    /* */

    numrows = 0
  )
  then
    raise_application_error(
      -20007,
      'Cannot UPDATE facturapublicacionpago because cuentabanco does not exist.'
    );
  end if;


  /* tipopago R/36 facturapublicacionpago ON CHILD UPDATE RESTRICT */
  select count(*) into numrows
    from tipopago
    where
      /* :new.tipopagocve = tipopago.tipopagocve */
      :new.tipopagocve = tipopago.tipopagocve;
  if (
    /* */

    numrows = 0
  )
  then
```

```
  raise_application_error(
    -20007,
    'Cannot UPDATE facturapublicacionpago because tipopago does not exist.'
  );
 end if;



end;
/

create trigger tD_proveedor after DELETE on proveedor for each row

-- DELETE trigger on proveedor
declare numrows INTEGER;
begin

   /* proveedor R/22 compra ON PARENT DELETE RESTRICT */
   select count(*) into numrows
     from compra
     where
       /*  compra.proveedorcve = :old.proveedorcve and
           compra.empresacve = :old.empresacve */
       compra.proveedorcve = :old.proveedorcve and
       compra.empresacve = :old.empresacve;
   if (numrows > 0)
   then
     raise_application_error(
       -20001,
       'Cannot DELETE proveedor because compra exists.'
     );
   end if;


   /* proveedor R/9 publicacion ON PARENT DELETE RESTRICT */
   select count(*) into numrows
     from publicacion
     where
       /*  publicacion.proveedorcve = :old.proveedorcve and
           publicacion.empresacve = :old.empresacve */
       publicacion.proveedorcve = :old.proveedorcve and
       publicacion.empresacve = :old.empresacve;
   if (numrows > 0)
   then
     raise_application_error(
       -20001,
       'Cannot DELETE proveedor because publicacion exists.'
     );
   end if;



end;
/

create trigger tI_proveedor after INSERT on proveedor for each row

-- INSERT trigger on proveedor
declare numrows INTEGER;
begin

   /* empresa R/18 proveedor ON CHILD INSERT RESTRICT */
```

```
  select count(*) into numrows
    from empresa
    where
      /* :new.empresacve = empresa.empresacve */
      :new.empresacve = empresa.empresacve;
  if (
    /*  */

    numrows = 0
  )
  then
    raise_application_error(
      -20002,
      'Cannot INSERT proveedor because empresa docs not exist.'
    );
  end if;



end;
/

create trigger tU_proveedor after UPDATE on proveedor for each row

-- UPDATE trigger on proveedor
declare numrows INTEGER;
begin

  /* proveedor R/22 compra ON PARENT UPDATE RESTRICT */
  if
    /* :old.proveedorcve <> :new.proveedorcve or
       :old.empresacve <> :new.empresacve */
    :old.proveedorcve <> :new.proveedorcve or
    :old.empresacve <> :new.empresacve
  then
    select count(*) into numrows
      from compra
      where
        /*  compra.proveedorcve = :old.proveedorcve and
            compra.empresacve = :old.empresacve */
        compra.proveedorcve = :old.proveedorcve and
        compra.empresacve = :old.empresacve;
    if (numrows > 0)
    then
      raise_application_error(
        -20005,
        'Cannot UPDATE proveedor because compra exists.'
      );
    end if;
  end if;


  /* proveedor R/9 publicacion ON PARENT UPDATE RESTRICT */
  if
    /* :old.proveedorcve <> :new.proveedorcve or
       :old.empresacve <> :new.empresacve */
    :old.proveedorcve <> :new.proveedorcve or
    :old.empresacve <> :new.empresacve
  then
    select count(*) into numrows
      from publicacion
      where
```

```
    /*  publicacion.proveedorcve = :old.proveedorcve and
        publicacion.empresacve = :old.empresacve */
    publicacion.proveedorcve = :old.proveedorcve and
    publicacion.empresacve = :old.empresacve;
  if (numrows > 0)
  then
   raise_application_error(
    -20005,
    'Cannot UPDATE proveedor because publicacion exists.'
   );
  end if;
 end if;


/* empresa R/18 proveedor ON CHILD UPDATE RESTRICT */
 select count(*) into numrows
  from empresa
  where
   /* :new.empresacve = empresa.empresacve */
   :new.empresacve = empresa.empresacve;
 if (
  /*  */

  numrows = 0
 )
 then
  raise_application_error(
   -20007,
   'Cannot UPDATE proveedor because empresa does not exist.'
  );
 end if;



end;
/

create trigger tD_publicacion after DELETE on publicacion for each row

-- DELETE trigger on publicacion
declare numrows INTEGER;
begin

   /* publicacion R/20 descuento ON PARENT DELETE RESTRICT */
   select count(*) into numrows
    from descuento
    where
     /*  descuento.publicacioncve = :old.publicacioncve and
        descuento.proveedorcve = :old.proveedorcve and
        descuento.empresacve = :old.empresacve */
     descuento.publicacioncve = :old.publicacioncve and
     descuento.proveedorcve = :old.proveedorcve and
     descuento.empresacve = :old.empresacve;
   if (numrows > 0)
   then
    raise_application_error(
     -20001,
     'Cannot DELETE publicacion because descuento exists.'
    );
   end if;
```

```
  /* publicacion R/10 edicion ON PARENT DELETE RESTRICT */
  select count(*) into numrows
    from edicion
    where
      /*  edicion.publicacioncve = :old.publicacioncve and
          edicion.proveedorcve = :old.proveedorcve and
          edicion.empresacve = :old.empresacve */
      edicion.publicacioncve = :old.publicacioncve and
      edicion.proveedorcve = :old.proveedorcve and
      edicion.empresacve = :old.empresacve;
  if (numrows > 0)
  then
    raise_application_error(
      -20001,
      'Cannot DELETE publicacion because edicion exists.'
    );
  end if;


end;
/

create trigger tI_publicacion after INSERT on publicacion for each row

-- INSERT trigger on publicacion
declare numrows INTEGER;
begin

  /* proveedor R/9 publicacion ON CHILD INSERT RESTRICT */
  select count(*) into numrows
    from proveedor
    where
      /* :new.proveedorcve = proveedor.proveedorcve and
         :new.empresacve = proveedor.empresacve */
      :new.proveedorcve = proveedor.proveedorcve and
      :new.empresacve = proveedor.empresacve;
  if (
    /*  */

    numrows = 0
  )
  then
    raise_application_error(
      -20002,
      'Cannot INSERT publicacion because proveedor does not exist.'
    );
  end if;


end;
/

create trigger tU_publicacion after UPDATE on publicacion for each row

-- UPDATE trigger on publicacion
declare numrows INTEGER;
begin

  /* publicacion R/20 descuento ON PARENT UPDATE RESTRICT */
  if
```

```
 /* :old.publicacioncve <> :new.publicacioncve or
    :old.proveedorcve <> :new.proveedorcve or
    :old.empresacve <> :new.empresacve */
  :old.publicacioncve <> :new.publicacioncve or
  :old.proveedorcve <> :new.proveedorcve or
  :old.empresacve <> :new.empresacve
then
  select count(*) into numrows
    from descuento
    where
      /* descuento.publicacioncve = :old.publicacioncve and
         descuento.proveedorcve = :old.proveedorcve and
         descuento.empresacve = :old.empresacve */
      descuento.publicacioncve = :old.publicacioncve and
      descuento.proveedorcve = :old.proveedorcve and
      descuento.empresacve = :old.empresacve;
  if (numrows > 0)
  then
    raise_application_error(
      -20005,
      'Cannot UPDATE publicacion because descuento exists.'
    );
  end if;
end if;


/* publicacion R/10 edicion ON PARENT UPDATE RESTRICT */
if
  /* :old.publicacioncve <> :new.publicacioncve or
     :old.proveedorcve <> :new.proveedorcve or
     :old.empresacve <> :new.empresacve */
  :old.publicacioncve <> :new.publicacioncve or
  :old.proveedorcve <> :new.proveedorcve or
  :old.empresacve <> :new.empresacve
then
  select count(*) into numrows
    from edicion
    where
      /* edicion.publicacioncve = :old.publicacioncve and
         edicion.proveedorcve = :old.proveedorcve and
         edicion.empresacve = :old.empresacve */
      edicion.publicacioncve = :old.publicacioncve and
      edicion.proveedorcve = :old.proveedorcve and
      edicion.empresacve = :old.empresacve;
  if (numrows > 0)
  then
    raise_application_error(
      -20005,
      'Cannot UPDATE publicacion because edicion exists.'
    );
  end if;
end if;


/* proveedor R/9 publicacion ON CHILD UPDATE RESTRICT */
select count(*) into numrows
  from proveedor
  where
    /* :new.proveedorcve = proveedor.proveedorcve and
       :new.empresacve = proveedor.empresacve */
    :new.proveedorcve = proveedor.proveedorcve and
    :new.empresacve = proveedor.empresacve;
```

```
 if (
  /* */

  numrows = 0
 )
 then
  raise_application_error(
   -20007,
   'Cannot UPDATE publicacion because proveedor does not exist.'
  );
 end if;



end;
/

create trigger tD_puntoentrega after DELETE on puntoentrega for each row

-- DELETE trigger on puntoentrega
declare numrows INTEGER;
begin

   /* puntoentrega R/61 rutageneradaoptima ON PARENT DELETE RESTRICT */
   select count(*) into numrows
    from rutageneradaoptima
    where
     /*  rutageneradaoptima.puntoentregacve = :old.puntoentregacve and
        rutageneradaoptima.rutacve = :old.rutacve and
        rutageneradaoptima.empleadocve = :old.empleadocve and
        rutageneradaoptima.departamentocve = :old.departamentocve and
        rutageneradaoptima.cargocve = :old.cargocve */
     rutageneradaoptima.puntoentregacve = :old.puntoentregacve and
     rutageneradaoptima.rutacve = :old.rutacve and
     rutageneradaoptima.empleadocve = :old.empleadocve and
     rutageneradaoptima.departamentocve = :old.departamentocve and
     rutageneradaoptima.cargocve = :old.cargocve;
   if (numrows > 0)
   then
    raise_application_error(
     -20001,
     'Cannot DELETE puntoentrega because rutageneradaoptima exists.'
    );
   end if;


   /* puntoentrega R/41 facturapublicacion ON PARENT DELETE RESTRICT */
   select count(*) into numrows
    from facturapublicacion
    where
     /*  facturapublicacion.puntoentregacve = :old.puntoentregacve and
        facturapublicacion.rutacve = :old.rutacve and
        facturapublicacion.empleadocve = :old.empleadocve and
        facturapublicacion.departamentocve = :old.departamentocve and
        facturapublicacion.cargocve = :old.cargocve */
     facturapublicacion.puntoentregacve = :old.puntoentregacve and
     facturapublicacion.rutacve = :old.rutacve and
     facturapublicacion.empleadocve = :old.empleadocve and
     facturapublicacion.departamentocve = :old.departamentocve and
     facturapublicacion.cargocve = :old.cargocve;
   if (numrows > 0)
   then
```

```
  raise_application_error(
   -20001,
   'Cannot DELETE puntoentrega because facturapublicacion exists.'
  );
 end if;


 /* puntoentrega R/14 clienteentrega ON PARENT DELETE RESTRICT */
 select count(*) into numrows
  from clienteentrega
  where
   /*  clienteentrega.puntoentregacve = :old.puntoentregacve and
      clienteentrega.rutacve = :old.rutacve and
      clienteentrega.empleadocve = :old.empleadocve and
      clienteentrega.departamentocve = :old.departamentocve and
      clienteentrega.cargocve = :old.cargocve */
    clienteentrega.puntoentregacve = :old.puntoentregacve and
    clienteentrega.rutacve = :old.rutacve and
    clienteentrega.empleadocve = :old.empleadocve and
    clienteentrega.departamentocve = :old.departamentocve and
    clienteentrega.cargocve = :old.cargocve;
 if (numrows > 0)
 then
  raise_application_error(
   -20001,
   'Cannot DELETE puntoentrega because clienteentrega exists.'
  );
 end if;



end;
/

create trigger tI_puntoentrega after INSERT on puntoentrega for each row

-- INSERT trigger on puntoentrega
declare numrows INTEGER;
begin

  /* ruta R/43 puntoentrega ON CHILD INSERT RESTRICT */
  select count(*) into numrows
   from ruta
   where
    /* :new.rutacve = ruta.rutacve and
      :new.empleadocve = ruta.empleadocve and
      :new.departamentocve = ruta.departamentocve and
      :new.cargocve = ruta.cargocve */
    :new.rutacve = ruta.rutacve and
    :new.empleadocve = ruta.empleadocve and
    :new.departamentocve = ruta.departamentocve and
    :new.cargocve = ruta.cargocve;
  if (
   /* */

   numrows = 0
  )
  then
   raise_application_error(
    -20002,
    'Cannot INSERT puntoentrega because ruta does not exist.'
   );
```

```
  end if;


end;
/

create trigger tU_puntoentrega after UPDATE on puntoentrega for each row

-- UPDATE trigger on puntoentrega
declare numrows INTEGER;
begin

 /* puntoentrega R/61 rutageneradaoptima ON PARENT UPDATE RESTRICT */
 if
  /* :old.puntoentregacve <> :new.puntoentregaeve or
     :old.rutacve <> :new.rutacve or
     :old.empleadocve <> :new.empleadocve or
     :old.departamentocve <> :new.departamentocve or
     :old.cargocve <> :new.cargocve */
   :old.puntoentregacve <> :new.puntoentregaeve or
   :old.rutacve <> :new.rutacve or
   :old.empleadocve <> :new.empleadocve or
   :old.departamentocve <> :new.departamentocve or
   :old.cargocve <> :new.cargocve
 then
  select count(*) into numrows
    from rutageneradaoptima
    where
     /* rutageneradaoptima.puntoentregacve = :old.puntoentregacve and
        rutageneradaoptima.rutacve = :old.rutacve and
        rutageneradaoptima.empleadocve = :old.empleadocve and
        rutageneradaoptima.departamentocve = :old.departamentocve and
        rutageneradaoptima.cargocve = :old.cargocve */
     rutageneradaoptima.puntoentregacve = :old.puntoentregacve and
     rutageneradaoptima.rutacve = :old.rutacve and
     rutageneradaoptima.empleadocve = :old.empleadocve and
     rutageneradaoptima.departamentocve = :old.departamentocve and
     rutageneradaoptima.cargocve = :old.cargocve;
  if (numrows > 0)
  then
   raise_application_error(
    -20005,
    'Cannot UPDATE puntoentrega because rutageneradaoptima exists.'
   );
  end if;
 end if;


 /* puntoentrega R/41 facturapublicacion ON PARENT UPDATE RESTRICT */
 if
  /* :old.puntoentregacve <> :new.puntoentregacve or
     :old.rutacve <> :new.rutacve or
     :old.empleadocve <> :new.empleadocve or
     :old.departamentocve <> :new.departamentocve or
     :old.cargocve <> :new.cargocve */
   :old.puntoentregacve <> :new.puntoentregacve or
   :old.rutacve <> :new.rutacve or
   :old.empleadocve <> :new.empleadocve or
   :old.departamentocve <> :new.departamentocve or
   :old.cargocve <> :new.cargocve
 then
```

```
 select count(*) into numrows
  from facturapublicacion
  where
   /*  facturapublicacion.puntocntregacve = :old.puntoentregacve and
      facturapublicacion.rutacve = :old.rutacve and
      facturapublicacion.empleadocve = :old.empleadocve and
      facturapublicacion.departamentocve = :old.departamentocve and
      facturapublicacion.cargocve = :old.cargocve */
   facturapublicacion.puntoentregacve = :old.puntoentregacve and
   facturapublicacion.rutacve = :old.rutacve and
   facturapublicacion.empleadocve = :old.empleadocve and
   facturapublicacion.departamentocve = :old.departamentocve and
   facturapublicacion.cargocve = :old.cargocve;
 if (numrows > 0)
 then
  raise_application_error(
   -20005,
   'Cannot UPDATE puntoentrega because facturapublicacion exists.'
  );
 end if;
end if;


/* puntoentrega R/14 clienteentrega ON PARENT UPDATE RESTRICT */
if
 /* :old.puntoentregacve <> :new.puntoentregacve or
   :old.rutacve <> :new.rutacve or
   :old.empleadocve <> :new.empleadocve or
   :old.departamentocve <> :new.departamentocve or
   :old.cargocve <> :new.cargocve */
 :old.puntocntregacve <> :new.puntoentregacve or
 :old.rutacve <> :new.rutacve or
 :old.empleadocve <> :new.empleadocve or
 :old.departamentocve <> :new.departamentocve or
 :old.cargocve <> :new.cargocve
then
 select count(*) into numrows
  from clienteentrega
  where
   /*  clienteentrega.puntoentregacve = :old.puntoentregacve and
      clienteentrega.rutacve = :old.rutacve and
      clienteentrega.empleadocve = :old.empleadocve and
      clienteentrega.departamentocve = :old.departamentocve and
      clienteentrega.cargocve = :old.cargocve */
   clienteentrega.puntoentregacve = :old.puntoentregacve and
   clienteentrega.rutacve = :old.rutacve and
   clienteentrega.empleadocve = :old.empleadocve and
   clienteentrega.departamentocve = :old.departamentocve and
   clienteentrega.cargocve = :old.cargocve;
 if (numrows > 0)
 then
  raise_application_error(
   -20005,
   'Cannot UPDATE puntoentrega because clienteentrega exists.'
  );
 end if;
end if;


/* ruta R/43 puntoentrega ON CHILD UPDATE RESTRICT */
select count(*) into numrows
 from ruta
```

```
  where
   /* :new.rutacve = ruta.rutacve and
      :new.empleadocve = ruta.empleadocve and
      :new.departamentocve = ruta.departamentocve and
      :new.cargocve = ruta.cargocve */
    :new.rutacve = ruta.rutacve and
    :new.empleadocve = ruta.empleadocve and
    :new.departamentocve = ruta.departamentocve and
    :new.cargocve = ruta.cargocve;
 if (
  /* */

  numrows = 0
 )
 then
  raise_application_error(
   -20007,
   'Cannot UPDATE puntoentrega because ruta does not exist.'
  );
 end if;



end;
/

create trigger tD_relacionembarque after DELETE on relacionembarque for each row

-- DELETE trigger on relacionembarque
declare numrows INTEGER;
begin

   /* relacionembarque R/52 facturadistribucion ON PARENT DELETE RESTRICT */
   select count(*) into numrows
     from facturadistribucion
     where
      /* facturadistribucion.relacionembarqueclave = :old.relacionembarqueclave and
         facturadistribucion.clientedifesacve = :old.clientedifesacve and
         facturadistribucion.tipoclientedifesacve = :old.tipoclientedifesacve and
         facturadistribucion.empresacve = :old.empresacve and
         facturadistribucion.situacionclientecve = :old.situacionclientecve and
         facturadistribucion.tipoclientecve = :old.tipoclientecve and
         facturadistribucion.tipoembarquecve = :old.tipoembarquecve */
       facturadistribucion.relacionembarqueclave = :old.relacionembarqueclave and
       facturadistribucion.clientedifesacve = :old.clientedifesacve and
       facturadistribucion.tipoclientedifesacve = :old.tipoclientedifesacve and
       facturadistribucion.empresacve = :old.empresacve and
       facturadistribucion.situacionclientecve = :old.situacionclientecve and
       facturadistribucion.tipoclientecve = :old.tipoclientecve and
       facturadistribucion.tipoembarquecve = :old.tipoembarquecve;
   if (numrows > 0)
   then
     raise_application_error(
       -20001,
       'Cannot DELETE relacionembarque because facturadistribucion exists.'
     );
   end if;


   /* relacionembarque R/46 almacenentradadistribucion ON PARENT DELETE RESTRICT */
   select count(*) into numrows
     from almacenentradadistribucion
```

```
    where
     /* almacenentradadistribucion.relacionembarqueclave = :old.relacionembarqueclave and
        almacenentradadistribucion.clientedifesacve = :old.clientedifesacve and
        almacenentradadistribucion.tipoclientedifesacve = :old.tipoclientedifesacve and
        almacenentradadistribucion.empresacve = :old.empresacve and
        almacenentradadistribucion.situacionclientecve = :old.situacionclientecve and
        almacenentradadistribucion.tipoclientecve = :old.tipoclientecve and
        almacenentradadistribucion.tipoembarquecve = :old.tipoembarquecve */
      almacenentradadistribucion.relacionembarqueclave = :old.relacionembarqueclave and
      almacenentradadistribucion.clientedifesacve = :old.clientedifesacve and
      almacenentradadistribucion.tipoclientedifesacve = :old.tipoclientedifesacve and
      almacenentradadistribucion.empresacve = :old.empresacve and
      almacenentradadistribucion.situacionclientecve = :old.situacionclientecve and
      almacenentradadistribucion.tipoclientecve = :old.tipoclientecve and
      almacenentradadistribucion.tipoembarquecve = :old.tipoembarquecve;
    if (numrows > 0)
    then
     raise_application_error(
      -20001,
      'Cannot DELETE relacionembarque because almacenentradadistribucion exists.'
     );
    end if;



end;
/

create trigger tI_relacionembarque after INSERT on relacionembarque for each row

-- INSERT trigger on relacionembarque
declare numrows INTEGER;
begin

    /* tipoembarque R/50 relacionembarque ON CHILD INSERT RESTRICT */
    select count(*) into numrows
     from tipoembarque
     where
      /* :new.tipoembarquecve = tipoembarque.tipoembarquecve */
      :new.tipoembarquecve = tipoembarque.tipoembarquecve;
    if (
     /* */

     numrows = 0
    )
    then
     raise_application_error(
      -20002,
      'Cannot INSERT relacionembarque because tipoembarque does not exist.'
     );
    end if;


    /* clientedifesa R/45 relacionembarque ON CHILD INSERT RESTRICT */
    select count(*) into numrows
     from clientedifesa
     where
      /* :new.clientedifesacve = clientedifesa.clientedifesacve and
         :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
         :new.empresacve = clientedifesa.empresacve and
         :new.situacionclientecve = clientedifesa.situacionclientecve and
         :new.tipoclientecve = clientedifesa.tipoclientecve */
```

```
    :new.clientedifesacve = clientedifesa.clientedifesacve and
    :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
    :new.empresacve = clientedifesa.cmpresacve and
    :new.situacionclientecve = clientedifesa.situacionclientecve and
    :new.tipoclientecve = clientedifesa.tipoclientecve;
  if (
   /* */

   numrows = 0
  )
  then
   raise_application_error(
    -20002,
    'Cannot INSERT relacionembarquc because clientedifesa does not exist.'
   );
  end if;



end;
/

create trigger tU_relacioncmbarque after UPDATE on relacionembarque for each row

-- UPDATE trigger on relacionembarque
declare numrows INTEGER;
begin

 /* relacionembarquc R/52 facturadistribucion ON PARENT UPDATE RESTRICT */
 if
  /* :old.relacionembarqueclave <> :new.relacionembarquecclave or
    :old.clientedifesacve <> :new.clientedifesacve or
    :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
    :old.empresacve <> :new.cmpresacve or
    :old.situacionclientecve <> :new.situacionclientecve or
    :old.tipoclientecve <> :new.tipoclientecve or
    :old.tipoembarquecve <> :new.tipoembarquecve */
  :old.relacionembarqueclave <> :new.relacionembarqueclave or
  :old.clientedifesacve <> :new.clientedifesacve or
  :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
  :old.empresacve <> :new.empresacve or
  :old.situacionclientecve <> :new.situacionclientecve or
  :old.tipoclientccve <> :new.tipoclientecve or
  :old.tipoembarquecve <> :new.tipoembarquecve
 then
  select count(*) into numrows
   from facturadistribucion
   where
    /* facturadistribucion.relacionembarqueclave = :old.relacionembarqueclave and
      facturadistribucion.clientedifesacve = :old.clientedifesacve and
      facturadistribucion.tipoclientedifesacve = :old.tipoclientedifesacve and
      facturadistribucion.empresacve = :old.empresacve and
      facturadistribucion.situacionclientecve = :old.situacionclientecve and
      facturadistribucion.tipoclientecve = :old.tipoclientecve and
      facturadistribucion.tipoembarquecve = :old.tipoembarquecve */
    facturadistribucion.relacionembarqueclave = :old.relacionembarqueclave and
    facturadistribucion.clientedifesacve = :old.clientedifesacve and
    facturadistribucion.tipoclientedifesacve = :old.tipoclientedifesacve and
    facturadistribucion.empresacve = :old.empresacve and
    facturadistribucion.situacionclientecve = :old.situacionclientecve and
    facturadistribucion.tipoclientecve = :old.tipoclientecve and
    facturadistribucion.tipoembarquecve = :old.tipoembarquecve;
```

```
 if (numrows > 0)
 then
  raise_application_error(
   -20005,
   'Cannot UPDATE relacionembarque because facturadistribucion exists.'
  );
 end if;
end if;


/* relacionembarque R/46 almacenentradadistribucion ON PARENT UPDATE RESTRICT */
if
 /* :old.relacionembarqueclave <> :new.relacionembarqueclave or
   :old.clientedifesacve <> :new.clientedifesacve or
   :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
   :old.empresacve <> :new.empresacve or
   :old.situacionclientecve <> :new.situacionclientecve or
   :old.tipoclientecve <> :new.tipoclientecve or
   :old.tipoembarquecve <> :new.tipoembarquecve */
 :old.relacionembarqueclave <> :new.relacionembarqueclave or
 :old.clientedifesacve <> :new.clientedifesacve or
 :old.tipoclientedifesacve <> :new.tipoclientedifesacve or
 :old.empresacve <> :new.empresacve or
 :old.situacionclientecve <> :new.situacionclientecve or
 :old.tipoclientecve <> :new.tipoclientecve or
 :old.tipoembarquecve <> :new.tipoembarquecve
then
 select count(*) into numrows
  from almacenentradadistribucion
  where
   /* almacenentradadistribucion.relacionembarqueclave = :old.relacionembarqueclave and
      almacenentradadistribucion.clientedifesacve = :old.clientedifesacve and
      almacenentradadistribucion.tipoclientedifesacve = :old.tipoclientedifesacve and
      almacenentradadistribucion.empresacve = :old.empresacve and
      almacenentradadistribucion.situacionclientecve = :old.situacionclientecve and
      almacenentradadistribucion.tipoclientecve = :old.tipoclientecve and
      almacenentradadistribucion.tipoembarquecve = :old.tipoembarquecve */
   almacenentradadistribucion.relacionembarqueclave = :old.relacionembarqueclave and
   almacenentradadistribucion.clientedifesacve = :old.clientedifesacve and
   almacenentradadistribucion.tipoclientedifesacve = :old.tipoclientedifesacve and
   almacenentradadistribucion.empresacve = :old.empresacve and
   almacenentradadistribucion.situacionclientecve = :old.situacionclientecve and
   almacenentradadistribucion.tipoclientecve = :old.tipoclientecve and
   almacenentradadistribucion.tipoembarquecve = :old.tipoembarquecve;
 if (numrows > 0)
 then
  raise_application_error(
   -20005,
   'Cannot UPDATE relacionembarque because almacenentradadistribucion exists.'
  );
 end if;
end if;


/* tipoembarque R/50 relacionembarque ON CHILD UPDATE RESTRICT */
select count(*) into numrows
 from tipoembarque
 where
  /* :new.tipoembarquecve = tipoembarque.tipoembarquecve */
  :new.tipoembarquecve = tipoembarque.tipoembarquecve;
if (
 /* */
```

```
  numrows = 0
)
then
  raise_application_error(
    -20007,
    'Cannot UPDATE relacionembarque because tipoembarque does not exist.'
  );
end if;


/* clientedifesa R/45 relacionembarque ON CHILD UPDATE RESTRICT */
select count(*) into numrows
  from clientedifesa
  where
    /* :new.clientedifesacve = clientedifesa.clientedifesacve and
       :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
       :new.empresacve = clientedifesa.empresacve and
       :new.situacionclientecve = clientedifesa.situacionclientecve and
       :new.tipoclientecve = clientedifesa.tipoclientecve */
    :new.clientedifesacve = clientedifesa.clientedifesacve and
    :new.tipoclientedifesacve = clientedifesa.tipoclientedifesacve and
    :new.empresacve = clientedifesa.empresacve and
    :new.situacionclientecve = clientedifesa.situacionclientecve and
    :new.tipoclientecve = clientedifesa.tipoclientecve;
if (
  /* */

  numrows = 0
)
then
  raise_application_error(
    -20007,
    'Cannot UPDATE relacionembarque because clientedifesa does not exist.'
  );
end if;



end;
/

create trigger tD_ruta after DELETE on ruta for each row

-- DELETE trigger on ruta
declare numrows INTEGER;
begin

    /* ruta R/62 bitacoraevento ON PARENT DELETE RESTRICT */
    select count(*) into numrows
      from bitacoraevento
      where
        /*  bitacoraevento.rutacve = :old.rutacve and
            bitacoraevento.empleadocve = :old.empleadocve and
            bitacoraevento.departamentocve = :old.departamentocve and
            bitacoraevento.cargocve = :old.cargocve */
        bitacoraevento.rutacve = :old.rutacve and
        bitacoraevento.empleadocve = :old.empleadocve and
        bitacoraevento.departamentocve = :old.departamentocve and
        bitacoraevento.cargocve = :old.cargocve;
    if (numrows > 0)
    then
```

```
  raise_application_error(
    -20001,
    'Cannot DELETE ruta because bitacoraevento exists.'
  );
 end if;


 /* ruta R/60 rutageneradaoptima ON PARENT DELETE RESTRICT */
 select count(*) into numrows
  from rutageneradaoptima
  where
    /*  rutageneradaoptima.rutacve = :old.rutacve and
        rutageneradaoptima.empleadocve = :old.empleadocve and
        rutageneradaoptima.departamentocve = :old.departamentocve and
        rutageneradaoptima.cargocve = :old.cargocve */
    rutageneradaoptima.rutacve = :old.rutacve and
    rutageneradaoptima.empleadocve = :old.empleadocve and
    rutageneradaoptima.departamentoeve = :old.departamentocve and
    rutageneradaoptima.cargocve = :old.cargocve;
 if (numrows > 0)
 then
  raise_application_error(
    -20001,
    'Cannot DELETE ruta because rutageneradaoptima exists.'
  );
 end if;


 /* ruta R/43 puntoentrega ON PARENT DELETE RESTRICT */
 select count(*) into numrows
  from puntoentrega
  where
    /*  puntoentrega.rutacve = :old.rutacve and
        puntoentrega.empleadocve = :old.empleadocve and
        puntoentrega.departamentocve = :old.departamentocve and
        puntoentrega.cargocve = :old.cargocve */
    puntoentrega.rutacve = :old.rutacve and
    puntoentrega.empleadocve = :old.empleadocve and
    puntoentrega.departamentocve = :old.departamentocve and
    puntoentrega.cargocve = :old.cargocve;
 if (numrows > 0)
 then
  raise_application_error(
    -20001,
    'Cannot DELETE ruta because puntoentrega exists.'
  );
 end if;



end;
/

create trigger tl_ruta after INSERT on ruta for each row

-- INSERT trigger on ruta
declare numrows INTEGER;
begin

  /* empleado R/42 ruta ON CHILD INSERT RESTRICT */
  select count(*) into numrows
    from empleado
```

```
    where
     /* :new.empleadocve = empleado.empleadocve and
        :new.departamentocve = empleado.departamentocve and
        :new.cargocve = empleado.cargocve */
      :new.empleadocve = empleado.empleadocve and
      :new.departamentocve = empleado.departamentocve and
      :new.cargocve = empleado.cargocve;
   if (
    /* */

    numrows = 0
   )
   then
    raise_application_error(
     -20002,
     'Cannot INSERT ruta because empleado does not exist.'
    );
   end if;



end;
/

create trigger tU_ruta after UPDATE on ruta for each row

-- UPDATE trigger on ruta
declare numrows INTEGER;
begin

 /* ruta R/62 bitacoraevento ON PARENT UPDATE RESTRICT */
 if
  /* :old.rutacve <> :new.rutacve or
     :old.empleadocve <> :new.empleadocve or
     :old.departamentocve <> :new.departamentocve or
     :old.cargocve <> :new.cargocve */
   :old.rutacve <> :new.rutacve or
   :old.empleadocve <> :new.empleadocve or
   :old.departamentocve <> :new.departamentocve or
   :old.cargocve <> :new.cargocve
  then
   select count(*) into numrows
    from bitacoraevento
    where
     /* bitacoraevento.rutacve = :old.rutacve and
        bitacoraevento.empleadocve = :old.empleadocve and
        bitacoraevento.departamentocve = :old.departamentocve and
        bitacoraevento.cargocve = :old.cargocve */
      bitacoraevento.rutacve = :old.rutacve and
      bitacoraevento.empleadocve = :old.empleadocve and
      bitacoraevento.departamentocve = :old.departamentocve and
      bitacoraevento.cargocve = :old.cargocve;
   if (numrows > 0)
   then
    raise_application_error(
     -20005,
     'Cannot UPDATE ruta because bitacoraevento exists.'
    );
   end if;
  end if;
```

```
/* ruta R/60 rutageneradaoptima ON PARENT UPDATE RESTRICT */
if
  /* :old.rutacve <> :new.rutacve or
     :old.empleadocve <> :new.empleadocve or
     :old.departamentocve <> :new.departamentocve or
     :old.cargocve <> :new.cargocve */
  :old.rutacve <> :new.rutacve or
  :old.empleadocve <> :new.empleadocve or
  :old.departamentocve <> :new.departamentocve or
  :old.cargocve <> :new.cargocve
then
  select count(*) into numrows
    from rutageneradaoptima
    where
      /* rutageneradaoptima.rutacve = :old.rutacve and
         rutageneradaoptima.empleadocve = :old.empleadocve and
         rutageneradaoptima.departamentocve = :old.departamentocve and
         rutageneradaoptima.cargocve = :old.cargocve */
      rutageneradaoptima.rutacve = :old.rutacve and
      rutageneradaoptima.empleadocve = :old.empleadocve and
      rutageneradaoptima.departamentocve = :old.departamentocve and
      rutageneradaoptima.cargocve = :old.cargocve;
  if (numrows > 0)
  then
    raise_application_error(
      -20005,
      'Cannot UPDATE ruta because rutageneradaoptima exists.'
    );
  end if;
end if;


/* ruta R/43 puntoentrega ON PARENT UPDATE RESTRICT */
if
  /* :old.rutacve <> :new.rutacve or
     :old.empleadocve <> :new.empleadocve or
     :old.departamentocve <> :new.departamentocve or
     :old.cargocve <> :new.cargocve */
  :old.rutacve <> :new.rutacve or
  :old.empleadocve <> :new.empleadocve or
  :old.departamentocve <> :new.departamentocve or
  :old.cargocve <> :new.cargocve
then
  select count(*) into numrows
    from puntoentrega
    where
      /* puntoentrega.rutacve = :old.rutacve and
         puntoentrega.empleadocve = :old.empleadocve and
         puntoentrega.departamentocve = :old.departamentocve and
         puntoentrega.cargocve = :old.cargocve */
      puntoentrega.rutacve = :old.rutacve and
      puntoentrega.empleadocve = :old.empleadocve and
      puntoentrega.departamentocve = :old.departamentocve and
      puntoentrega.cargocve = :old.cargocve;
  if (numrows > 0)
  then
    raise_application_error(
      -20005,
      'Cannot UPDATE ruta because puntoentrega exists.'
    );
  end if;
end if;
```

```
/* empleado R/42 ruta ON CHILD UPDATE RESTRICT */
select count(*) into numrows
  from empleado
  where
   /* :new.empleadocve = empleado.empleadocve and
      :new.departamentocve = empleado.departamentocve and
      :new.cargocve = empleado.cargocve */
    :new.empleadocve = empleado.empleadocve and
    :new.departamentocve = empleado.departamentocve and
    :new.cargocve = empleado.cargocve;
if (
  /* */

  numrows = 0
)
then
  raise_application_error(
    -20007,
    'Cannot UPDATE ruta because empleado does not exist.'
  );
end if;



end;
/

create trigger tI_rutageneradaoptima after INSERT on rutageneradaoptima for each row

-- INSERT trigger on rutageneradaoptima
declare numrows INTEGER;
begin

  /* puntoentrega R/61 rutageneradaoptima ON CHILD INSERT RESTRICT */
  select count(*) into numrows
    from puntoentrega
    where
     /* :new.puntoentregacve = puntoentrega.puntoentregacve and
        :new.rutacve = puntoentrega.rutacve and
        :new.empleadocve = puntoentrega.empleadocve and
        :new.departamentocve = puntoentrega.departamentocve and
        :new.cargocve = puntoentrega.cargocve */
      :new.puntoentregacve = puntoentrega.puntoentregacve and
      :new.rutacve = puntoentrega.rutacve and
      :new.empleadocve = puntoentrega.empleadocve and
      :new.departamentocve = puntoentrega.departamentocve and
      :new.cargocve = puntoentrega.cargocve;
  if (
    /* */

    numrows = 0
  )
  then
    raise_application_error(
      -20002,
      'Cannot INSERT rutageneradaoptima because puntoentrega does not exist.'
    );
  end if;
```

```
  /* ruta R/60 rutageneradaoptima ON CHILD INSERT RESTRICT */
  select count(*) into numrows
    from ruta
    where
      /* :new.rutacve = ruta.rutacve and
         :new.empleadocve = ruta.empleadocve and
         :new.departamentocve = ruta.departamentocve and
         :new.cargocve = ruta.cargocve */
      :new.rutacve = ruta.rutacve and
      :new.empleadocve = ruta.empleadocve and
      :new.departamentocve = ruta.departamentocve and
      :new.cargocve = ruta.cargocve;
  if (
    /* */

    numrows = 0
  )
  then
    raise_application_error(
      -20002,
      'Cannot INSERT rutageneradaoptima because ruta does not exist.'
    );
  end if;



end;
/

create trigger tU_rutageneradaoptima after UPDATE on rutageneradaoptima for each row

-- UPDATE trigger on rutageneradaoptima
declare numrows INTEGER;
begin

  /* puntoentrega R/61 rutageneradaoptima ON CHILD UPDATE RESTRICT */
  select count(*) into numrows
    from puntoentrega
    where
      /* :new.puntoentregacve = puntoentrega.puntoentregacve and
         :new.rutacve = puntoentrega.rutacve and
         :new.empleadocve = puntoentrega.empleadocve and
         :new.departamentocve = puntoentrega.departamentocve and
         :new.cargocve = puntoentrega.cargocve */
      :new.puntoentregacve = puntoentrega.puntoentregacve and
      :new.rutacve = puntoentrega.rutacve and
      :new.empleadocve = puntoentrega.empleadocve and
      :new.departamentocve = puntoentrega.departamentocve and
      :new.cargocve = puntoentrega.cargocve;
  if (
    /* */

    numrows = 0
  )
  then
    raise_application_error(
      -20007,
      'Cannot UPDATE rutageneradaoptima because puntoentrega does not exist.'
    );
  end if;
```

```
/* ruta R/60 rutageneradaoptima ON CHILD UPDATE RESTRICT */
select count(*) into numrows
  from ruta
  where
   /* :new.rutacve = ruta.rutacve and
      :new.empleadocve = ruta.empleadocve and
      :new.departamentocve = ruta.departamentocve and
      :new.cargocve = ruta.cargocve */
    :new.rutacve = ruta.rutacve and
    :new.empleadocve = ruta.empleadocve and
    :new.departamentocve = ruta.departamentocve and
    :new.cargocve = ruta.cargocve;
if (
  /* */

  numrows = 0
)
then
  raise_application_error(
    -20007,
    'Cannot UPDATE rutageneradaoptima because ruta does not exist.'
  );
end if;



end;
/

create trigger tD_situacioncliente after DELETE on situacioncliente for each row

-- DELETE trigger on situacioncliente
declare numrows INTEGER;
begin

   /* situacioncliente R/12 clientedifesa ON PARENT DELETE RESTRICT */
   select count(*) into numrows
     from clientedifesa
     where
       /* clientedifesa.situacionclientecve = :old.situacionclientecve */
       clientedifesa.situacionclientecve = :old.situacionclientecve;
   if (numrows > 0)
   then
     raise_application_error(
       -20001,
       'Cannot DELETE situacioncliente because clientedifesa exists.'
     );
   end if;



end;
/

create trigger tU_situacioncliente after UPDATE on situacioncliente for each row

-- UPDATE trigger on situacioncliente
declare numrows INTEGER;
begin

  /* situacioncliente R/12 clientedifesa ON PARENT UPDATE RESTRICT */
  if
```

```
  /* :old.situacionclientecve <> :new.situacionclientecve */
  :old.situacionclientecve <> :new.situacionclientecve
 then
  select count(*) into numrows
   from clientedifesa
   where
    /* clientedifesa.situacionclientecve = :old.situacionclientecve */
    clientedifesa.situacionclientecve = :old.situacionclientecve;
  if (numrows > 0)
  then
   raise_application_error(
    -20005,
    'Cannot UPDATE situacioncliente because clientedifesa exists.'
   );
  end if;
 end if;



end;
/

create trigger tD_tipocliente after DELETE on tipocliente for each row

-- DELETE trigger on tipocliente
declare numrows INTEGER;
begin

  /* tipocliente R/13 clientedifesa ON PARENT DELETE RESTRICT */
  select count(*) into numrows
   from clientedifesa
   where
    /* clientedifesa.tipoclientecve = :old.tipoclientecve */
    clientedifesa.tipoclientecve = :old.tipoclientecve;
  if (numrows > 0)
  then
   raise_application_error(
    -20001,
    'Cannot DELETE tipocliente because clientedifesa exists.'
   );
  end if;



end;
/

create trigger tU_tipocliente after UPDATE on tipocliente for each row

-- UPDATE trigger on tipocliente
declare numrows INTEGER;
begin

 /* tipocliente R/13 clientedifesa ON PARENT UPDATE RESTRICT */
 if
  /* :old.tipoclientecve <> :new.tipoclientecve */
  :old.tipoclientecve <> :new.tipoclientecve
 then
  select count(*) into numrows
   from clientedifesa
   where
    /* clientedifesa.tipoclientecve = :old.tipoclientecve */
```

```
    clientedifesa.tipoclientecve = :old.tipoclientecve;
  if (numrows > 0)
  then
   raise_applieation_error(
    -20005,
    'Cannot UPDATE tipocliente because clientedifesa exists.'
   );
  end if;
 end if;



end;
/

create trigger tD_tipoembarque after DELETE on tipoembarque for each row

-- DELETE trigger on tipoembarque
declare numrows INTEGER;
begin

  /* tipoembarque R/54 distribucioncosto ON PARENT DELETE RESTRICT */
  select count(*) into numrows
   from distribucioncosto
   where
    /*  distribucioncosto.tipoembarquecve = :old.tipoembarquecve */
    distribucioncosto.tipoembarquecve = :old.tipoembarquecve;
  if (numrows > 0)
  then
   raise_application_error(
    -20001,
    'Cannot DELETE tipoembarque because distribucioncosto exists.'
   );
  end if;


  /* tipoembarque R/50 relacionembarque ON PARENT DELETE RESTRICT */
  select count(*) into numrows
   from relacionembarque
   where
    /*  relacionembarque.tipoembarquecve = :old.tipoembarquecve */
    relacionembarque.tipoembarquecve = :old.tipoembarquecve;
  if (numrows > 0)
  then
   raise_application_error(
    -20001,
    'Cannot DELETE tipoembarque because relacionembarque exists.'
   );
  end if;



end;
/

create trigger tU_tipoembarque after UPDATE on tipoembarque for each row

-- UPDATE trigger on tipoembarque
declare numrows INTEGER;
begin

 /* tipoembarque R/54 distribucioncosto ON PARENT UPDATE RESTRICT */
```

```
if
 /* :old.tipoembarquecve <> :new.tipoembarquecve */
 :old.tipoembarquecve <> :new.tipoembarquecve
then
 select count(*) into numrows
  from distribucioncosto
  where
   /* distribucioncosto.tipoembarquecve = :old.tipoembarquecve */
   distribucioncosto.tipoembarquecve = :old.tipoembarquecve;
 if (numrows > 0)
 then
  raise_application_error(
   -20005,
   'Cannot UPDATE tipoembarque because distribucioncosto exists.'
  );
 end if;
end if;


/* tipoembarque R/50 relacionembarque ON PARENT UPDATE RESTRICT */
if
 /* :old.tipoembarquecve <> :new.tipoembarquecve */
 :old.tipoembarquecve <> :new.tipoembarquecve
then
 select count(*) into numrows
  from relacionembarque
  where
   /* relacionembarque.tipoembarquecve = :old.tipoembarquecve */
   relacionembarque.tipoembarquecve = :old.tipoembarquecve;
 if (numrows > 0)
 then
  raise_application_error(
   -20005,
   'Cannot UPDATE tipoembarque because relacionembarque exists.'
  );
 end if;
end if;



end;
/

create trigger tD_tipoevento after DELETE on tipoevento for each row

-- DELETE trigger on tipoevento
declare numrows INTEGER;
begin

  /* tipoevento R/63 bitacoraevento ON PARENT DELETE RESTRICT */
  select count(*) into numrows
   from bitacoraevento
   where
    /* bitacoraevento.tipoeventocve = :old.tipoeventocve */
    bitacoraevento.tipoeventocve = :old.tipoeventocve;
  if (numrows > 0)
  then
   raise_application_error(
    -20001,
    'Cannot DELETE tipoevento because bitacoraevento exists.'
   );
  end if;
```

```
end;
/

create trigger tU_tipoevento after UPDATE on tipoevento for each row

-- UPDATE trigger on tipoevento
declare numrows INTEGER;
begin

 /* tipoevento R/63 bitacoraevento ON PARENT UPDATE RESTRICT */
 if
  /* :old.tipoeventocve <> :new.tipoeventocve */
  :old.tipoeventocve <> :new.tipoeventocve
 then
  select count(*) into numrows
   from bitacoraevento
   where
    /* bitacoraevento.tipoeventocve = :old.tipoeventocve */
    bitacoraevento.tipoeventocve = :old.tipoeventocve;
  if (numrows > 0)
  then
   raise_application_error(
    -20005,
    'Cannot UPDATE tipoevento because bitacoraevento exists.'
   );
  end if;
 end if;



end;
/

create trigger tD_tipopago after DELETE on tipopago for each row

-- DELETE trigger on tipopago
declare numrows INTEGER;
begin

  /* tipopago R/58 facturadistribucionpago ON PARENT DELETE RESTRICT */
  select count(*) into numrows
   from facturadistribucionpago
   where
    /* facturadistribucionpago.tipopagocve = :old.tipopagocve */
    facturadistribucionpago.tipopagocve = :old.tipopagocve;
  if (numrows > 0)
  then
   raise_application_error(
    -20001,
    'Cannot DELETE tipopago because facturadistribucionpago exists.'
   );
  end if;


  /* tipopago R/36 facturapublicacionpago ON PARENT DELETE RESTRICT */
  select count(*) into numrows
   from facturapublicacionpago
   where
    /* facturapublicacionpago.tipopagocve = :old.tipopagocve */
```

```
    facturapublicacionpago.tipopagocve = :old.tipopagocve;
  if (numrows > 0)
  then
   raise_application_error(
    -20001,
    'Cannot DELETE tipopago because facturapublicacionpago exists.'
   );
  end if;


  /* tipopago R/29 comprapago ON PARENT DELETE RESTRICT */
  select count(*) into numrows
   from comprapago
   where
    /*  comprapago.tipopagocve = :old.tipopagocve */
    comprapago.tipopagocve = :old.tipopagocve;
  if (numrows > 0)
  then
   raise_application_error(
    -20001,
    'Cannot DELETE tipopago because comprapago exists.'
   );
  end if;



end;
/

create trigger tU_tipopago after UPDATE on tipopago for each row

-- UPDATE trigger on tipopago
declare numrows INTEGER;
begin

 /* tipopago R/58 facturadistribucionpago ON PARENT UPDATE RESTRICT */
 if
  /* :old.tipopagocve <> :new.tipopagocve */
  :old.tipopagocve <> :new.tipopagocve
 then
  select count(*) into numrows
   from facturadistribucionpago
   where
    /*  facturadistribucionpago.tipopagocve = :old.tipopagocve */
    facturadistribucionpago.tipopagocve = :old.tipopagocve;
  if (numrows > 0)
  then
   raise_application_error(
    -20005,
    'Cannot UPDATE tipopago because facturadistribucionpago exists.'
   );
  end if;
 end if;


 /* tipopago R/36 facturapublicacionpago ON PARENT UPDATE RESTRICT */
 if
  /* :old.tipopagocve <> :new.tipopagocve */
  :old.tipopagocve <> :new.tipopagocve
 then
  select count(*) into numrows
   from facturapublicacionpago
```

```
    where
     /* facturapublicacionpago.tipopagocve = :old.tipopagocve */
     facturapublicacionpago.tipopagocve = :old.tipopagocve;
   if (numrows > 0)
   then
    raise_application_error(
     -20005,
     'Cannot UPDATE tipopago because facturapublicacionpago exists.'
    );
   end if;
  end if;


  /* tipopago R/29 comprapago ON PARENT UPDATE RESTRICT */
  if
   /* :old.tipopagocve <> :new.tipopagocve */
   :old.tipopagocve <> :new.tipopagocve
  then
   select count(*) into numrows
    from comprapago
    where
     /* comprapago.tipopagocve = :old.tipopagocve */
     comprapago.tipopagocve = :old.tipopagocve;
   if (numrows > 0)
   then
    raise_application_error(
     -20005,
     'Cannot UPDATE tipopago because comprapago exists.'
    );
   end if;
  end if;



end;
/

create trigger tD_tipousuario after DELETE on tipousuario for each row

-- DELETE trigger on tipousuario
declare numrows INTEGER;
begin

   /* tipousuario R/16 usuario ON PARENT DELETE RESTRICT */
   select count(*) into numrows
    from usuario
    where
     /* usuario.tipousuariocve = :old.tipousuariocve */
     usuario.tipousuariocve = :old.tipousuariocve;
   if (numrows > 0)
   then
    raise_application_error(
     -20001,
     'Cannot DELETE tipousuario because usuario exists.'
    );
   end if;



end;
/
```

```
create trigger tU_tipousuario after UPDATE on tipousuario for each row

-- UPDATE trigger on tipousuario
declare numrows INTEGER;
begin

 /* tipousuario R/16 usuario ON PARENT UPDATE RESTRICT */
 if
  /* :old.tipousuariocve <> :new.tipousuariocve */
  :old.tipousuariocve <> :new.tipousuariocve
 then
  select count(*) into numrows
   from usuario
   where
    /*  usuario.tipousuariocve = :old.tipousuariocve */
    usuario.tipousuariocve = :old.tipousuariocve;
  if (numrows > 0)
  then
   raise_application_error(
    -20005,
    'Cannot UPDATE tipousuario because usuario exists.'
   );
  end if;
 end if;



end;
/

create trigger tI_usuario after INSERT on usuario for each row

-- INSERT trigger on usuario
declare numrows INTEGER;
begin

  /* empleado R/17 usuario ON CHILD INSERT RESTRICT */
  select count(*) into numrows
   from empleado
   where
    /* :new.empleadocve = empleado.empleadocve and
       :new.departamentocve = empleado.departamentocve and
       :new.cargocve = empleado.cargocve */
     :new.empleadocve = empleado.empleadocve and
     :new.departamentocve = empleado.departamentocve and
     :new.cargocve = empleado.cargocve;
  if (
   /* */

   numrows = 0
  )
  then
   raise_application_error(
    -20002,
    'Cannot INSERT usuario because empleado does not exist.'
   );
  end if;


  /* tipousuario R/16 usuario ON CHILD INSERT RESTRICT */
  select count(*) into numrows
   from tipousuario
```

```
    where
      /* :new.tipousuariocve = tipousuario.tipousuariocve */
      :new.tipousuariocve = tipousuario.tipousuariocve;
  if (
    /* */

    numrows = 0
  )
  then
    raise_application_error(
      -20002,
      'Cannot INSERT usuario because tipousuario does not exist.'
    );
  end if;



end;
/

create trigger tU_usuario after UPDATE on usuario for each row

-- UPDATE trigger on usuario
declare numrows INTEGER;
begin

  /* empleado R/17 usuario ON CHILD UPDATE RESTRICT */
  select count(*) into numrows
    from empleado
    where
      /* :new.empleadocve = empleado.empleadocve and
        :new.departamentocve = empleado.departamentocve and
        :new.cargocve = empleado.cargocve */
      :new.empleadocve = empleado.empleadocve and
      :new.departamentocve = empleado.departamentocve and
      :new.cargocve = empleado.cargocve;
  if (
    /* */

    numrows = 0
  )
  then
    raise_application_error(
      -20007,
      'Cannot UPDATE usuario because empleado does not exist.'
    );
  end if;


  /* tipousuario R/16 usuario ON CHILD UPDATE RESTRICT */
  select count(*) into numrows
    from tipousuario
    where
      /* :new.tipousuariocve = tipousuario.tipousuariocve */
      :new.tipousuariocve = tipousuario.tipousuariocve;
  if (
    /* */

    numrows = 0
  )
  then
    raise_application_error(
```

```
    -20007,
    'Cannot UPDATE usuario because tipousuario does not exist.'
  );
 end if;


end;
/
```

```
/* FIN DE TESIS*/
   RETURN (1);
```