76

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

SISTEMA PARA LA DISTRIBUCION Y CONTROL DE UNIFORMES Y ROPA DE TRABAJO EN EL SISTEMA DE TRANSPORTE COLECTIVO (METRO)

T E S I S

QUE PARA OBTENER EL TITULO DE:

INGENIERO EN COMPUTACION

P R E S E N T A:

MONICA RAMIREZ HERNANDEZ

DIRECTOR DE TESIS:

ING. MARICELA CASTAÑEDA PERDOMO

MEXICO, D. F. NOVIEMBRE. 2000

A la Ing. Maricela Castañeda Perdomo:
Por su confianza, paciencia y sabiduría que amablemente compartio conmigo.

A la Facultad de Ingeniería:
Por haberme formado como Ingeniero.

A la Universidad Nacional Autónoma de México:
Por abrirme sus puertas y haberme dado la oportunidad de llegar a ser una profesionista.

Mil Gracias...

Mónica

A mis padres, hermanos y quienes estuvierón conmigo para que este sueño se hiciera realidad:

Como un testimonio de mi infinito aprecio y agradecimiento
por toda una vida de esfuerzos y sacrificios
brindandome siempre cariño y apoyo
cuando más lo necesite,
deseo con todo el corazón que mi triunfo
como mujer y profesionista
lo sientan como el suyo propio.

Con amor, admiración y respeto...

Mónica

INDICE

CAPITULO 1. INTRODUCCION

En la actualidad, los sistemas de información son el corazón de diversas actividades cotidianas y objeto de gran consideración en la toma de decisiones.

Por tal motivo hubo la necesidad de realizar El Sistema de Uniformes y Ropa de Trabajo para el S.T.C. (Sistema de Transporte Colectivo (Metro)), el cual proporciona la dotación al personal que por la naturaleza de las funciones que realiza lo requiere para su seguridad y buena presentación.

En el desarrollo de este sistema se tomó en cuenta a todos aquellos que hacen uso de las aplicaciones, es decir, los usuarios finales y se estudian cada una de las actividades asociadas a dicho sistema, además de considerar los siguientes puntos:

* Se dará trámite a solicitudes extemporáneas, nuevo ingreso y cambio de categoría o adscripción.

* Todos los trabajadores tienen derecho, siempre que la categoría a la que pertenecen se encuentre en el catálogo.

Para el desarrollo se tomaron en cuenta las siguientes fases establecidas por el usuario.

1A. FASE

1. - ACTUALIZACION A CATALOGOS

Se realiza el mantenimiento (altas, bajas, cambios, consultas) a los catálogos principales del sistema los cuales son: Catálogo de Prendas, Catálogo de Categorías y Adscripciones y Archivo Maestro de Ropa.

2. - GENERACION DE MAESTRO DE ROPA

El archivo maestro de ropa se genera en base al catálogo de categorías y adscripciones y el archivo maestro de empleados actualizado. Este proceso se realiza catorcenalmente.

3. - INTEGRACION DE EXCEPCIONES

Existen trabajadores que la categoría o departamento al que pertenecen no está calificada o se realiza un cambio a la dotación que le corresponde, sin embargo existe una autorización para que le sea proporcionada cierta dotación, por lo tanto se debe integrar en el archivo maestro de ropa identificando al trabajador como excepción.

4. - GENERACION DE CEDULAS DE RECOPILACION DE TALLAS

Por medio de un conducto (sindicato o empresa) se le hará llegar al trabajador la cédula de recopilación de tallas, en donde proporcionará las tallas correctas

A través de la Gerencia de Recursos Humanos se recabarán las tallas de los trabajadores.

5. - CAPTURA DE TALLAS

Una vez que se recopilan las cédulas de tallas se procede a su captura, incluyendo reportes de validaciones para realizar las correcciones correspondientes.

2A. FASE

1. - GENERACION DE CEDULAS DE ENTREGA

A través del sindicato (personal base) o de la empresa (personal de confianza) se le hace llegar al trabajador su cédula con la cual se procederá a la entrega de la dotación que le corresponde por medio de la Gerencia de Recursos Materiales, la cual se basará en un calendario por área de adscripción.

2. - CAPTURA DE PRENDAS ENTREGADAS

Existen diferentes circunstancias por las cuales se realiza la entrega de dotación parcial, por lo que es importante llevar un control de la cantidad de prendas que se han entregado al trabajador.

3A. FASE

1. - INFORMACION PARA PRESUPUESTO

Obtener la información necesaria para elaborar la requisición de uniformes y ropa de trabajo respectiva.

4A. FASE

1. - REPORTES

Se incluyen diversos reportes los cuales son de gran utilidad para el manejo y buen funcionamiento del sistema.

Las fases descritas anteriormente sirvieron como base para establecer la forma en que se haría el diseño.

Con respecto al Marco de Referencia se hace mención a todo lo que implico realizar este sistema: con que se contaba, que se habia realizado, problemas que se tenian, sus posibles soluciones y cómo finalmente se toma la decisión de realizarlo.

Por otra parte, en la Proyección del Sistema se hace notar su alcance, los recursos en base a los cuales se desarrollo, así como el análisis de los costos y beneficios que éste traerá al S.T.C. (Metro). Se examina también la utilidad que tiene en la empresa y se estudian tres pruebas de factibilidad: operacional, técnica y financiera.

El análisis y diseño al igual que otros temas de no menor importancia están contenidos dentro de la Metodología para el Desarrollo del Sistema. Con el análisis se definieron las características y alcance funcional, así como también los requerimientos en base a especificaciones establecidas por el usuario. Con el diseño conceptual se hizo posible tener la idea clara de lo que éste representaría, se estimaron tanto los recursos como los tiempos necesarios para un desarrollo adecuado. En la programación se pusieron en práctica los estándares propios del lenguaje y los establecidos para el desarrollo. En la documentación se definieron prototipos tanto para el manual técnico como para el manual de usuario. La implementación se hizo en donde iba a trabajar el sistema checando que todo quedara bien. La capacitación se realizó operativa y técnicamente apoyada en los manuales de usuario y técnico y por último en la liberación se verá cómo se entregó oficialmente el sistema. Finalmente se encuentran las conclusiones.

CAPITULO 2. MARCO DE REFERENCIA

2.1 ANTECEDENTES

El Sistema de Transporte Colectivo (Metro) otorga ropa y/o calzado a los empleados que tienen derecho a la prestación. Por este motivo, se tenía un sistema cuya herramienta de desarrollo era un lenguaje de 2a. generación (Cobol) y funcionaba en un equipo 36 de IBM que tiene las siguientes características: 512 kbytes de memoria, 600 Mbytes en disco, dos impresoras 600 lpm, una unidad de cinta y doce terminales.

El usuario tenía muchos problemas con dicho sistema, debido a esto se vio la necesidad de desarrollar uno nuevo que cumpliera al 100% con sus requerimientos.

Para solucionar este problema se pensó inicialmente en un desarrollo con programación de un lenguaje de tercera generación (C, Pascal), instalado en una PC, el cual cubriera las necesidades generales.

Al analizar más a detalle se notó la falta de compatibilidad entre la PC y el equipo en el cual se tenían archivos requeridos para un buen funcionamiento, además de que se iban a necesitar tres impresoras a la hora de la generación de solicitudes para los empleados, ya que son alrededor de 12,500 las que se imprimirían y una impresora no se daría abasto, por consiguiente el sistema tendría que estar instalado en tres PC´s. Con respecto al tiempo sería muy grande el que se tendría que invertir en dicha tarea.

En fechas anteriores el equipo 36 de IBM fue sustituido por un equipo HP-9000 cuyas características son: 128 Mbytes de memoria RAM, 6 Gbytes en disco duro, dos impresoras 1100 lpm, una unidad de cartucho y soporta hasta 32 terminales.

Al tomar en cuenta que se contaba con el equipo HP-9000 surgió una segunda propuesta que consistía en desarrollar un sistema en dicho equipo, en lenguaje de 4ta. generación totalmente operativo, con procedimientos optimizados y su respectiva documentación tanto técnica como de usuario.

Se notó que esta opción era la más adecuada porque se podrían cubrir todos los requerimientos, el tiempo empleado en los procesos y en la impresión sería el necesario y el usuario podría tomar el control total después de recibir una capacitación correspondiente.

Para el desarrollo se cuenta con un sistema operativo multiusuario UNIX (HP-UX).

Al ser INFORMIX 4GL RDS (Rapid Development System) la herramienta seleccionada para el desarrollo, se vio que esta aplica procedimientos para compilar y ejecutar programas 4GL eliminando la necesidad de utilizar compiladores de lenguaje C o ligas de C lo que reduce significativamente el tiempo de compilación.

Con SQL (Structured Query Language) todo el equipo de trabajo utilizaría los nuevos conceptos informáticos relacionados con la base de datos relacional.

Con relación a la presentación de pantallas, todos los menús serían del tipo windows, donde las opciones se presentarían en la parte superior de la pantalla y el usuario podría seleccionarlas desplazándose con las flechas.

Esta segunda propuesta de solución fue autorizada y con respecto al tiempo de desarrollo éste fue de cuatro meses y medio aproximadamente.

La realización de este proyecto no hubiese podido realizarse sin la valiosa colaboración de los usuarios ya que siempre estuvieron dispuestos a ayudar cuando se les necesitaba.

Afortunadamente los tiempos para la liberación del sistema se cumplieron y los requerimientos del usuario se cubrieron en un 100%, dando opción al desarrollo de nuevos aplicaciones propias del sistema.

2.2 PROBLEMATICA

El problema surge cuando el sistema que actualmente se utiliza no cubre las necesidades del usuario.

Los problemas más relevantes que se notaron se enuncian a continuación:

1. - No existía flexibilidad, ya que si se necesitaba alguna información especial, el sistema constantemente se estaba modificando para obtener la información solicitada.

2. - La información era demasiado generalizada, para obtener totales de las prendas por sexo, departamento y/o categoría el usuario tenía que consultar la plantilla de personal autorizada para separar manualmente las cantidades y verificar que checaran realmente los totales que da el sistema con los que se obtenían por separado.

3. - La información dada por el sistema no era confiable, el usuario tenía que checar todo manualmente, por lo tanto ocupaban dicha información sólo como referencia.

4. - En determinada época, no se tenía acceso a cierta información y eso detenía el ciclo de trabajo.

5. - El usuario no manejaba el sistema, sino que requería de la ayuda de los responsables de éste para obtener su información.

6. - Los procesos eran lentos y en ocasiones se tenían que realizar tareas manualmente por falta de tiempo, ya que no podían esperar hasta que el sistema les proporcionara la información.

7. - Los reportes que se entregaban eran exageradamente grandes, en ocasiones se tenían que unir dos o más reportes para saber con exactitud la información requerida.

8. - Para realizar una consulta a algún empleado, había que buscar manualmente o pedirle al responsable del sistema que le proporcionara lo que necesitaba, cosa que en muchas ocasiones era negada por estar desarrollando alguna otra actividad.

9. - No se tenía documentación, motivo por el cual se desconocía totalmente su funcionamiento y era difícil cambiar de responsables puesto que tampoco tenían manual técnico.

Todos estos problemas ocasionaron que el usuario estuviera descontento, no confiaba en la información que se le daba y prefería trabajar manualmente que intentar realizar alguna tarea en el sistema, motivo por el cual tuvo que desarrollarse uno nuevo que cubriera al 100% sus requerimientos.

CAPITULO 3. PROYECCION DEL SISTEMA

3.1 ALCANCE Y RECURSOS

La finalidad de un sistema es la razón de su existencia; dicha finalidad es procesar entradas, mantener archivos de datos relacionados con la organización y producir información, reportes y otras salidas.

Para tal acción los recursos que se tienen para la realización del Sistema son:

- El equipo de cómputo en que se trabaja es un HP-9000 cuyas características se mencionaron en la introducción.
- Se cuenta con un sistema operativo multiusuario UNIX (HP-UX) y la herramienta seleccionada es INFORMIX 4GL RDS (Rapid Development System).
- Se contó con el apoyo de dos usuarios que expresaron sus requerimientos por medio de los cuales pudimos ver cual era el alcance de este sistema.

El aspecto más importante es la experiencia humana y el empleo de ideas para aprovechar la computadora con la finalidad de que éstas lleven a cabo las tareas necesarias. Para que se tuviera el alcance deseado y fuera utilizado, fue esencial la comunicación con los usuarios ya que el éxito de la implementación dependió de la capacidad de comunicación en forma significativa.

El alcance es que debe ser fácil de utilizar y adecuarse a la organización para la que fue diseñado, ya que si ayuda al usuario a trabajar con mayor eficiencia entonces éste lo utilizará, de lo contrario, lo evitará, como esto último es lo que menos se desea, se deberán entregar oportunamente los uniformes y ropa que a cada trabajador le corresponde, llevar un control total del área para operar en un 100%, disminuir los tiempos y eliminar las tareas que se realizaban manualmente.

Según los requerimientos del usuario se podían cubrir en un 100% sus necesidades reduciendo el tiempo invertido para la realización de su trabajo.

3.2 ANALISIS COSTO/BENEFICIO

El Análisis Costo/Beneficio es una evaluación de la justificación económica para un proyecto basado en computadora.

Para maximizar la utilidad de la información, se debe manejar correctamente, tal como se deben manejar los demás recursos. Se necesita comprender que hay costos asociados con la producción, distribución, seguridad, almacenamiento y recuperación de toda información.

Aunque la información se encuentra a nuestro alrededor esta no es gratis, y su uso es estratégico para posicionar la competitividad de un sistema.

En realidad el análisis de costo/beneficio no se realizo sobre el equipo, puesto que ya se contaba con el y no se quería invertir dinero en comprar hardware ni software.

Los costos que se tomaron en cuenta para la realización de este sistema son:

- Costo de personal dedicado a la realización del sistema.
- Costo para el mantenimiento del sistema.
- Costo de los alquileres (teléfono, electricidad)
- Costo de la plantilla involucrada en las actividades de gestión, operación y planificación del sistema.
- Costo de papelería

Para el Sistema de Transporte Colectivo (Metro) los costos fueron poco significativos, ya que la persona que realizó este sistema, esta integrada en la plantilla de personal, sólo se tuvo que capacitar a la persona que se haría cargo del mantenimiento. Debido a que las personas involucradas para uso del sistema se encuentran en la estación del metro Isabel la Católica y el sistema estar implantado en el metro Zaragoza el costo del teléfono se ve involucrado ya que el usuario hará uso de éste para saber algunas cosas del sistema mientras llega a trabajar en dicho lugar.

Respecto al costo de la electricidad, debido a que se tiene una red local para trabajar, el gasto de energía eléctrica que producirá el sistema es relativamente pequeño y no se invirtió mucho tiempo en dicho costo porque el S.T.C. (Metro) globaliza los gastos de energía eléctrica y teléfono. Solo se hace notar que se hará uso de dichos elementos para que el sistema funcionara adecuadamente.

Hay reducción en los costos de papelería, ya que la cantidad de papel que se utilizaba con el sistema anterior era demasiado grande comparada con la que se utilizará ahora.

El costo en tiempo de las personas involucradas en la realización de este proyecto no se vio afectado, puesto que el usuario intervino en su horario laboral y muy

esporádicamente tuvo que quedarse fuera de éste a menos que fueran causas de fuerza mayor, además de que el encargado estaba dedicado en un 100%.

Los beneficios del sistema se determinaron de acuerdo con el modo de trabajo ya existente.

Dichos beneficios fueron:

- Reducción del costo además de un incremento en la velocidad de cálculos e impresiones.
- Mejora la exactitud de las tareas de cálculo.
- Posibilidad de cambiar rápidamente las variables y los valores en los programas.
- Posibilidad de recoger y guardar automáticamente datos de los registros.
- Mantenimiento de los registros más completo y más sistemático.
- Aumento de la capacidad para el mantenimiento de registros en términos de espacio.
- Estandarización del mantenimiento de registros.
- Aumento de la cantidad de datos que se pueden guardar por registro.
- Mejora la seguridad en el almacenamiento de registros, posibilidades de acceso y de cambio en base de datos.
- Obtención de registros más rápida.
- Posibilidad de cambiar simultáneamente clases enteras de registros.
- Posibilidad de crear nuevos archivos, mezclando partes de otros archivos.
- Reducción de la necesidad de trabajo forzado en el control de procesos y de recursos.
- Posibilidad de agregar grandes cantidades de datos de distintas formas que sean útiles para la planificación y la toma de decisiones.

Al observar que los costos eran menores a los beneficios que podíamos obtener, se tuvo la aprobación por parte del S.T.C. (Metro) para el desarrollo de este sistema.

3.3 ESTUDIO DE FACTIBILIDAD

Fue necesario y prudente evaluar la viabilidad porque se vio que se pudieron evitar semanas o meses de esfuerzo, dinero y una inversión profesional incalculable que hubieran afectado el ciclo de vida del sistema.

Se realizó el análisis costo/beneficio mencionado anteriormente, y el estudio de funcionalidad, así como el rendimiento y las restricciones que pudieran afectar a que el sistema fuera aceptable.

Debido al manejador de base de datos a usar y al estar dividido por módulos era funcional, además de tener un muy buen rendimiento ya que disminuía tiempo de inversión con respecto al usuario. La restricción que tendría es solo que al generar el archivo maestro que contendrá la información con la que se trabajara, si ya se hubiesen capturado las tallas, se perderían dichas tallas si el trabajador cambiaba de categoría, puesto que la dotación de ropa para cada categoría es diferente y el usuario tendría que volver a capturar sus tallas en la nueva dotación.

Se estudio también el hecho de que no tuviera ninguna infracción, violación o ilegalidad al ser desarrollado y posteriormente liberado, por esta parte no se tuvo ningún problema.

La factibilidad se valoro de tres formas diferentes:

Factibilidad Técnica:

Una gran parte de la determinación de recursos tuvo que ver con la valoración de la factibilidad técnica. Se vio que los recursos técnicos que se tenían para el desarrollo de este sistema eran viables, no se necesito mejorarles o añadirles nada puesto que satisfacían la petición bajo consideración.

Factibilidad económica:

Esta fue la segunda parte de la determinación de recursos. Los recursos básicos a considerar fueron: el tiempo propio y el equipo de sistemas. Se estudiaron en el análisis de costo/beneficio los costos posibles, dando por conclusión que si era viable ya que los costos a corto plazo serían sobrepasados por las ganancias a largo plazo y producirían una reducción inmediata en los costos de operación.

Factibilidad operacional:

La factibilidad operacional dependería de los recursos humanos disponibles para el proyecto, e involucraría proyectar si el sistema operara y será usado una vez que este instalado. Al pedir que fuera eficiente y accesible, y que el desarrollo cumpliría esta petición, se pudo notar que iba a operar en un 100%, puesto que reduciría el tiempo de intervención y sería confiable la información proporcionada.

Con lo anterior se puede observar que:

El estudio del sistema se logro rápidamente, con el fin de que los recursos que se le dedicaran fueran mínimos, la información producida por el estudio fuera sólida y cualquier interés existente en el proyecto se mantuviera alto.

Debido a que este estudio es preliminar al análisis, fue ejecutado en forma rápida y competente.

En resumen podemos decir que la factibilidad significa que el proyecto propuesto:

- Ayuda a que la organización logre sus objetivos generales.
- Es posible de lograr con los recursos actuales de la organización en las siguientes tres áreas:

Factibilidad técnica.

Tecnología disponible para satisfacer las necesidades de los usuarios.

Factibilidad económica.

El tiempo del analista de sistemas.

Costo del estudio de sistemas.

Costo del tiempo de los empleados para el estudio.

Factibilidad operacional.

Si el sistema trabajará cuando sea instalado.

Si el sistema será usado.

Para dar por finalizada esta etapa y que el usuario diera la decisión de seguir o no seguir se entregó un documento que contenía lo siguiente:

- Declaración del problema
- Entorno de la implementación
- Restricciones
- Hallazgos importantes
- Comentarios
- Recomendaciones
- Viabilidad el sistema

Al revisar el usuario dicho documento y que decidió que si se podía seguir con la realización se notó que las metas y los requisitos se pueden expresar en términos de atributos de calidad que se deben poseer y se tomó como base el árbol de la figura 3.1 para analizar si era viable y con calidad.

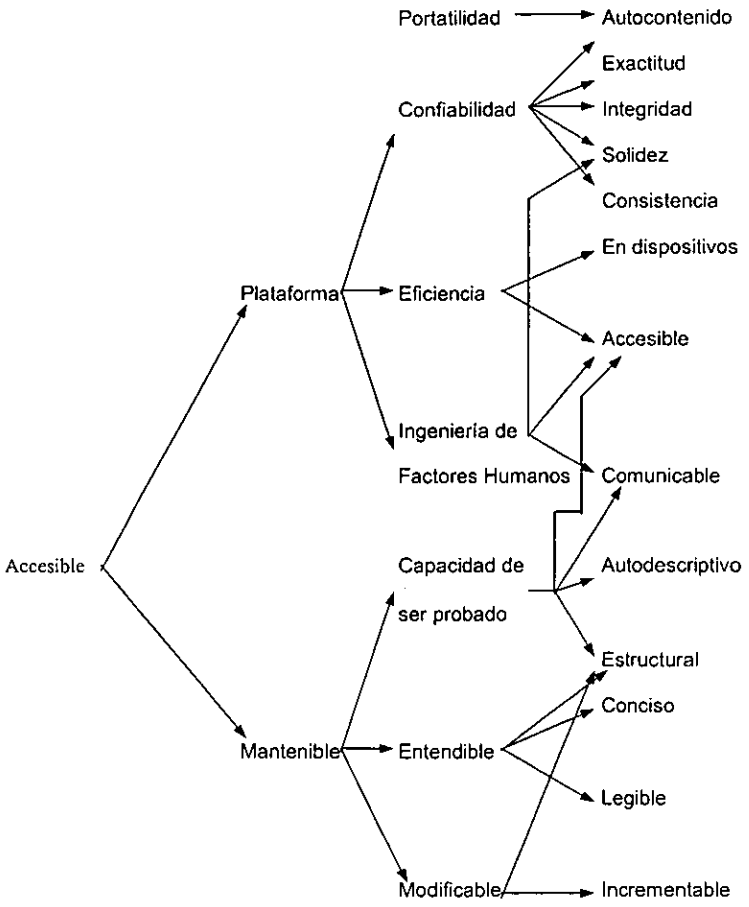


Figura 3.1 Arbol de factibilidad

Con todo lo anterior podemos decir que el sistema tuvo la aprobación del S.T.C. (Metro) para su desarrollo, así como el consentimiento del usuario para su realización.

CAPITULO 4.

METODOLOGIA PARA EL DESARROLLO DEL SISTEMA

4.1 HERRAMIENTAS PARA EL ANALISIS

A medida que fluye por un sistema basado en computadora, la información se transforma. El sistema acepta entradas en una gran variedad de formas, aplica los elementos de hardware, software y humanos para transformar la entrada en salida y produce salida en una gran variedad de formas. El análisis estructurado es una técnica de modelización del flujo y del contenido de la información. En la figura 4.1 el sistema basado en computadora se representa como una transformación de información. Se representa el funcionamiento general del sistema como una única transformación de información que en la figura aparece como una burbuja. En entidades externas representadas por cuadros, se originan una o más entradas que aparecen como flechas etiquetadas. La entrada conduce la transformación que produce información de salida (también representadas como flechas etiquetadas) dirigida hacia otras entidades externas. Podemos señalar que este modelo se puede aplicar al sistema completo o solamente al elemento de software. La clave esta en representar la información que entra y la que es producida por la transformación.

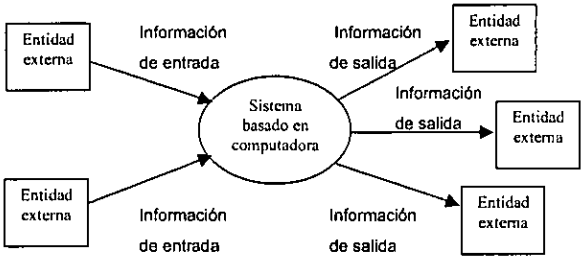


Figura 4.1 Modelo del flujo de información

4.1.1 EL DIAGRAMA DE FLUJO DE DATOS Y SUS CARACTERISTICAS

A medida que la información se mueve a través del software, es modificada por una serie de transformaciones. El diagrama de flujo de datos (DFD) es una técnica gráfica que representa el flujo de la información y las transformaciones que se aplican a los datos al moverse desde la entrada hasta la salida. También es conocido como grafo de flujo de datos o como diagrama de burbujas.

Se emplea para describir y analizar el movimiento de datos a través del sistema, y es la herramienta más importante y la base sobre la cual se desarrollan otros componentes.

CARACTERISTICAS GENERALES DE LOS DFD'S

- Cualquier flujo de datos que abandone un proceso debe estar basado en los datos que entran al proceso.
- Todos los flujos de datos reciben un nombre, el nombre refleja los datos que fluyen entre procesos, almacenes de datos, fuentes o destinos.
- Sólo deben entrar al proceso los datos necesarios para llevarlo a cabo.
- Un proceso no debe saber nada de ningún otro en el sistema, es decir, debe ser independiente; la única dependencia que debe existir es aquella que este basada en sus propios datos de entrada y salida.
- Los procesos siempre están en continua ejecución; no se inician ni tampoco se detienen, siempre están listos para funcionar o realizar el trabajo necesario.
- La salida de los procesos puede tomar una de las siguientes formas:
- Flujo de datos con información añadida por el proceso.
- Una respuesta o cambio en la forma de los datos.
- Un cambio de condición.
- Un cambio de contenido.
- Cambios en la organización, es decir, reacomodo de datos.

En el apéndice A se muestra una parte de los diagramas.

4.1.2 EL DICCIONARIO DE DATOS

El diccionario de datos es un listado organizado de todos los elementos de datos que son pertinentes para el sistema, con definiciones precisas y rigurosas que permiten que el usuario y el analista tengan una misma comprensión del mismo. Es un componente importante del análisis, ya que proporciona más información relacionada con el sistema.

Es un catálogo, un depósito, de los elementos en un sistema. Como su nombre lo sugiere, estos elementos se centran alrededor de los datos y la forma en que están estructurados para satisfacer los requerimientos de los usuarios y las necesidades de la organización. En el diccionario de datos se encuentra la lista de todos los elementos que forman parte del flujo de datos de todo el sistema.

IMPORTANCIA DEL DICCIONARIO DE DATOS

- Manejar los detalles.

El sistema tiene un gran volumen de datos que fluyen por él en forma de documentos, reportes, etc. Todo sistema experimenta cambios continuos y manejar de manera completa todos los detalles es un desafío. Es imposible que se recuerde todo y si se trata de hacer lo más seguro es que se cometan invariablemente equivocaciones o se olviden elementos importantes, por tal motivo se registra la información utilizando un diccionario de datos automatizado diseñado de manera especifica para el análisis y diseño del sistema.

- Comunicar un significado común para todos los elementos.

El diccionario de datos proporciona asistencia para asegurar significados comunes para los elementos y actividades del sistema.

- Documentar las características.

Tener la descripción formal de las características produce una comprensión más completa. Una vez que las características estén articuladas y registradas, se tendrá una fuente común de información con respecto al sistema.

- Facilitar el análisis de los detalles con la finalidad de evaluar las características y determinar donde efectuar cambios.

Se tiene que determinar si son necesarias nuevas características o si están en orden los cambios de cualquier tipo.

- Localizar errores y omisiones.

Tener información en un diccionario relacionada con las características dice mucho con respecto al sistema y permite evaluarlo. Pero para esto es necesario saber que la propia información es completa y exacta. Por consiguiente los diccionarios se emplean para localizar errores en la descripción.

Además de proporcionar documentación y eliminar redundancia, el diccionario de datos puede ser usado para:

- Validar el diagrama de flujo de datos y para confirmar que este completo y preciso.
- Proporcionar un punto inicial para el desarrollo de pantallas y reportes.
- Determinar el contenido de datos almacenados en archivos.
- Desarrollar la lógica para los diagramas de flujo de datos de procesos.

En el apéndice B se muestra el diccionario de datos.

4.1.3 LAS MINIESPECIFICACIONES Y METODOS PARA SU REALIZACION

Seguir procedimientos y tomar decisiones son aspectos importantes; por tal motivo se examinaron varias herramientas para el estudio de procedimientos de operación y de los pasos a seguir para la toma de decisiones junto con los medios para documentar estos aspectos en el estudio. Una herramienta es cualquier dispositivo objeto u operación utilizada para ejecutar una tarea especifica.

Se presentan aquí herramientas para documentar los procedimientos (Se tomo el programa rop08m el cual genera el archivo maestro de ropa para ejemplificar dichas herramientas).

4.1.3.1 ARBOLES DE DECISION

El árbol de decisión es un diagrama que representa en forma secuencial condiciones y acciones; muestra que condiciones se consideran en primer lugar, cuales en segundo y así sucesivamente. Ese método también permite mostrar la relación que existe entre cada condición y el grupo de acciones permisibles asociado con ella.

Los diagramas de este tipo se parecen a las raíces de un árbol, de aquí su nombre (figura 4.2).

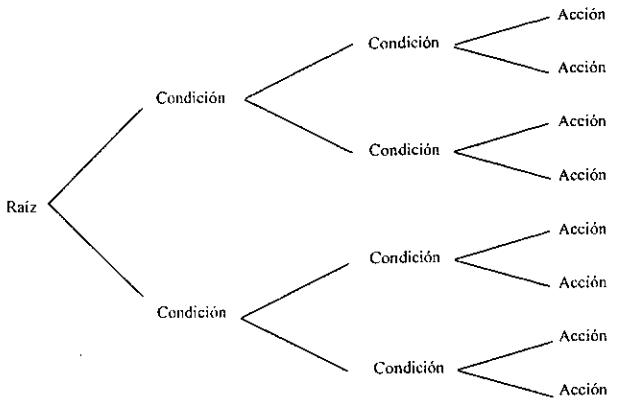


Figura 4.2 La secuencia de decisiones en un árbol de decisión es de izquierda a derecha

La raíz del árbol que aparece en la parte izquierda de la figura, es el punto donde comienza la secuencia de decisión. La rama a seguir depende de las condiciones existentes y de la decisión que debe tomarse. Al avanzar de izquierda a derecha por una rama en particular, se obtiene una serie de toma de decisiones. Después de cada punto de decisión, se encuentra el siguiente conjunto de decisiones a considerar. De esta forma, los nodos del árbol representan condiciones y señalan la necesidad de tomar una determinación relacionada con la existencia de alguna de éstas, antes de seleccionar la siguiente trayectoria. La parte que se encuentra a la derecha del árbol indica las acciones que deben realizarse, las que a su vez dependen de la secuencia de condiciones que las preceden.

En la Figura 4.3 se muestra el árbol del programa

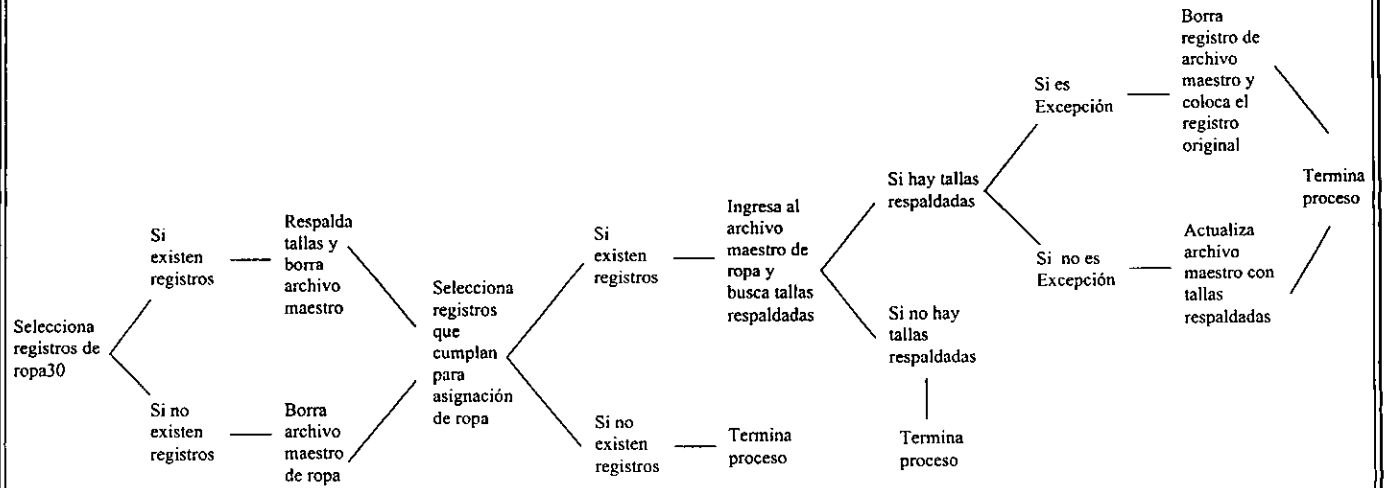


Figura 4.3 Arbol de decisión del programa rop08m.

4.1.3.2 TABLAS DE DECISION

Mas que un árbol, la tabla de decisión es una matriz de renglones y columnas que indican condiciones y acciones. Las reglas de decisión, incluidas en una tabla de decisión, establecen el procedimiento de seguir cuando existen ciertas condiciones. Este método se emplea desde mediados de la década de los 50's.

La tabla de decisión esta integrada por cuatro secciones: identificación de condiciones, entradas de condiciones, identificación de acciones y entradas de acciones (figura 4.4).

La identificación de condiciones señala aquellas que son relevantes. Las entradas de condiciones indican que valor, si es que lo hay, se debe asociar para una determinada condición. La identificación de acciones enlista el conjunto de todos los pasos que se deben seguir cuando se presenta cierta condición. Las entradas de acciones muestran las acciones especificas del conjunto que deben emprenderse cuando ciertas condiciones o combinaciones de éstas son verdaderas. En ocasiones se añaden notas en la parte interior de la tabla para indicar cuando utilizar la tabla o para diferenciarla de otras tablas de decisión.

Las columnas del lado derecho de la tabla enlazan condiciones y acciones, forman reglas de decisión que establecen las condiciones que deben satisfacerse para emprender un determinado conjunto de acciones.

Nótese que se omite el orden de la secuencia, cosa que no sucede con los árboles de decisión.

La regla de decisión incorpora todas las condiciones que deben ser ciertas y no sólo una a la vez.

CONDICIONES	REGLAS DE DECISION
Identificación de condiciones	Entrada de acciones
Identificación de acciones	Entrada de condiciones

Figura 4.4 Forma general de las tablas de decisión

Pasos para construir una tabla de decisión.

1. Determinar los factores considerados como más relevantes en la toma de decisiones. Esto permite identificar las condiciones en la decisión. Cada condición seleccionada debe tener la característica de ocurrir o no ocurrir, en este caso no es posible la ocurrencia parcial.

2. Determinar los pasos o actividades más factibles bajo condiciones que cambian. Esto permite identificar las acciones.

3. Estudiar las diferentes posibilidades de combinaciones de condiciones.

4. Llenar la tabla con las reglas de decisión.

5. Marcar las entradas correspondientes a las acciones con una X para indicar que éstas se emprenden; dejar las celdas vacías o marcadas con un guión para señalar que en ese renglón no se emprende ninguna acción.

6. Examinar la tabla para detectar reglas redundantes o contradicciones entre éstas.

ELIMINACION DE LA REDUNDANCIA

Las tablas de decisión pueden volverse muy grandes y difíciles de manejar si se permite que crezcan sin ningún control. Remover las entradas redundantes puede ser de ayuda para manejar el tamaño de la tabla. La redundancia se presenta cuando las siguientes condiciones son verdaderas al mismo tiempo.

SUPRESION DE CONTRADICCIONES

Las reglas de decisión son contradictorias entre si, cuando dos o más reglas tienen el mismo conjunto de condiciones pero sus acciones son diferentes.

Las contradicciones indican que la información que se tiene es incorrecta o bien que existe un error en la construcción de la tabla. Sin embargo, muchas veces la contradicción es resultado de las discrepancias en la información que se recibe de diferentes personas con respecto a la forma en que éstas toman decisiones. Se puede tomar una decisión especifica utilizando diferentes reglas. Encontrar tales discrepancias es de gran utilidad para poder mejorar una situación de decisión.

En la figura 4.5 se muestra la tabla de decisión del programa.

REGLAS DE DECISION								
CONDICION	Respalda tallas y borra archivo maestro de ropa	Borra archivo maestro de ropa	Ingresa al archivo maestro de ropa y busca tallas respaldadas	Verifica si hay tallas respaldadas	Verifica si es excepción	Borra registro del archivo maestro y coloca el registro original	Actualiza archivo maestro con tallas respaldadas	Termina proceso
Selecciona registros								
SI existen registros	X		X	X			X	
No existen registros		X	X			X	X	
Selecciona registros que cumplan para asignación de ropa								
Si existen registros			X	X			X	
No existen registros								X
Si hay tallas respaldadas					X			
Si no hay tallas respaldadas								X
Si es excepción						X		X
Si no es excepción							X	X

Figura 4.5 Tabla de decisión del programa rop08m

4.1.4 ESPAÑOL ESTRUCTURADO

El español estructurado es otro método para evitar los problemas de ambigüedad del lenguaje al establecer condiciones y acciones, tanto en procedimientos como en decisiones.

Este método no hace uso de árboles o tablas; en su lugar utiliza declaraciones para describir el proceso. El método no muestra las reglas de decisión; las declara.

Aún con esta característica, las especificaciones en español estructurado requieren que se identifiquen primero las condiciones que se presentan en un proceso y las decisiones que se deben tomar cuando esto sucede, junto con las acciones correspondientes. Sin embargo, este método también permite hacer una lista de todas los pasos en el orden en que se llevan a cabo. Para ello no se utilizan símbolos ni formatos especiales, características de los árboles y tablas de decisión, que a veces resultan incómodos. Además, es posible describir con rapidez los procedimientos en su totalidad ya que para ello se emplean declaraciones muy similares al español.

La terminología utilizada en la descripción estructurada de una aplicación consiste, en gran medida, en nombres de datos para los elementos que están definidos en el diccionario de datos.

DESARROLLO DE DECLARACIONES ESTRUCTURADAS

El español estructurado emplea tres tipos básicos de declaraciones para describir un proceso: estructuras de secuencia, estructuras de decisión y estructuras de iteración. Estas estructuras son adecuadas para el análisis de decisión y pueden trasladarse al desarrollo de software y programación.

ESTRUCTURAS DE SECUENCIA

Una estructura de secuencia es un solo paso o acción incluido en un proceso. Esto no depende de la existencia de ninguna condición y, cuando se encuentra, siempre se lleva a cabo. En general, se emplean varias instrucciones en secuencia para describir un proceso.

ESTRUCTURAS DE DECISION

El español estructurado es otro camino para mostrar el análisis de decisión. Por tanto, a menudo se incluyen las secuencias de acciones dentro de estructuras de decisión que sirven para identificar condiciones. Es así como las estructuras de decisión aparecen cuando se pueden emprender dos o más acciones. Lo que depende del valor de una condición específica.

Para esto, primero se evalúa la condición y después se toma la decisión de emprender las acciones o el grupo de acciones asociado con esta condición. Una vez determinada la condición las acciones son incondicionales.

Las estructuras de decisión no están limitadas a pares de combinaciones. Pueden existir muchas condiciones. La figura 4.6 muestra la anidación de varios niveles de condiciones y acciones para cada punto de decisión.

ESTRUCTURAS DE ITERACION

En las actividades rutinarias de operación, es común encontrar que algunas de ellas se repiten mientras existen ciertas condiciones o hasta que éstas se presentan. Las instrucciones de iteración permiten describir estos casos.

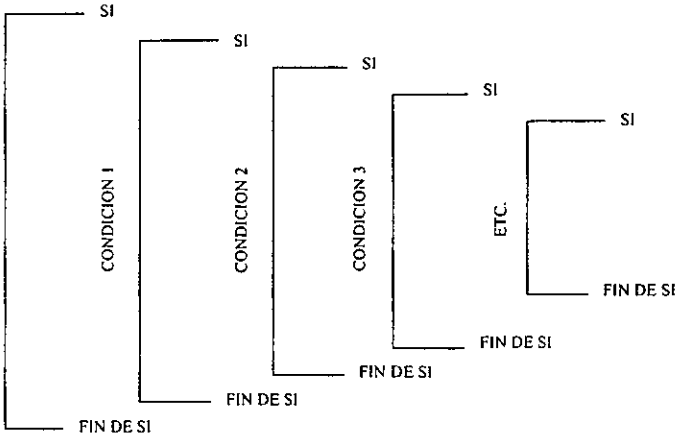


Figura 4.6 Anidación de estructuras de decisión en especificaciones en español estructurado

En la figura 4.7 se muestra el español estructurado del programa.

Selecciona registros de ropa30

Si existen registros

    Respalda tallas y borra archivo maestro de ropa

Si no existen registros

    Borra archivo maestro de ropa

Selecciona registros que cumplan para asignación de ropa

Si existen registros

    Ingresa al archivo maestro de ropa y busca tallas respaldadas

    Si existen tallas respaldadas

        Si es excepción

            Borra registro del archivo maestro de ropa y coloca el registro original

            Termina proceso

        Si no es excepción

            Actualiza archivo maestro con tallas respaldadas

            Termina proceso

    Si no existen tallas respaldadas

        Termina proceso

Si no existen registros

    Termina proceso

Figura 4.7 Español estructurado del programa rop08m

4.2 DISEÑO DEL SISTEMA

El diseño es realmente un proceso que se centra en cuatro atributos distintos de un programa: estructura de datos, arquitectura del software, representaciones de interfaz y algoritmo.

El proceso de diseño traduce requisitos en una representación del software que se pueda evaluar por calidad antes de que comience la generación de código. Al igual que los requisitos, el diseño se documenta y se hace parte de la documentación del software.

4.2.1 LA GRAFICA DE ESTRUCTURA Y SUS CARACTERISTICAS

La gráfica de estructura es una herramienta de diseño que muestra gráficamente las relaciones entre los módulos de un programa. Presenta cuáles módulos interactuan dentro del sistema y también muestra gráficamente los datos que se comunican entre varios módulos.

Por conveniencia y facilidad de comunicación se uso una simbología común en la elaboración de diagramas de estructura. Los módulos del programa se identifican mediante rectángulos (figura 4.8), con el nombre del módulo escrito dentro del rectángulo. Las flechas indican las llamadas, las cuales son cualquier mecanismo para invocar un módulo en particular.

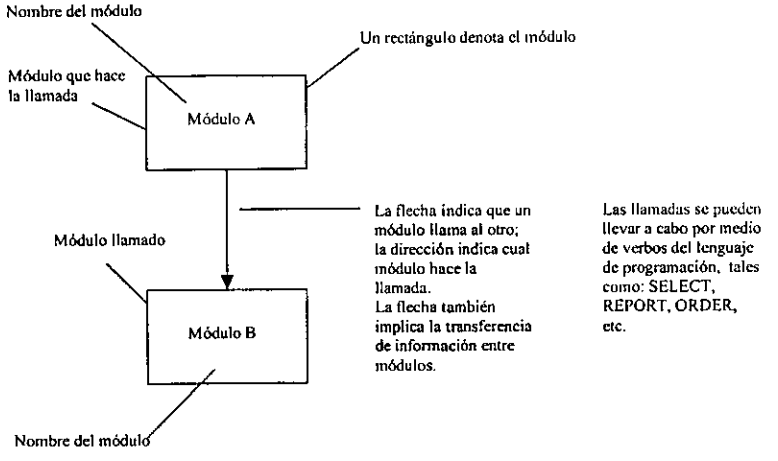


Figura 4.8 Notación usada en gráficas de estructura

En la figura 4.9 se muestra la gráfica de estructura.

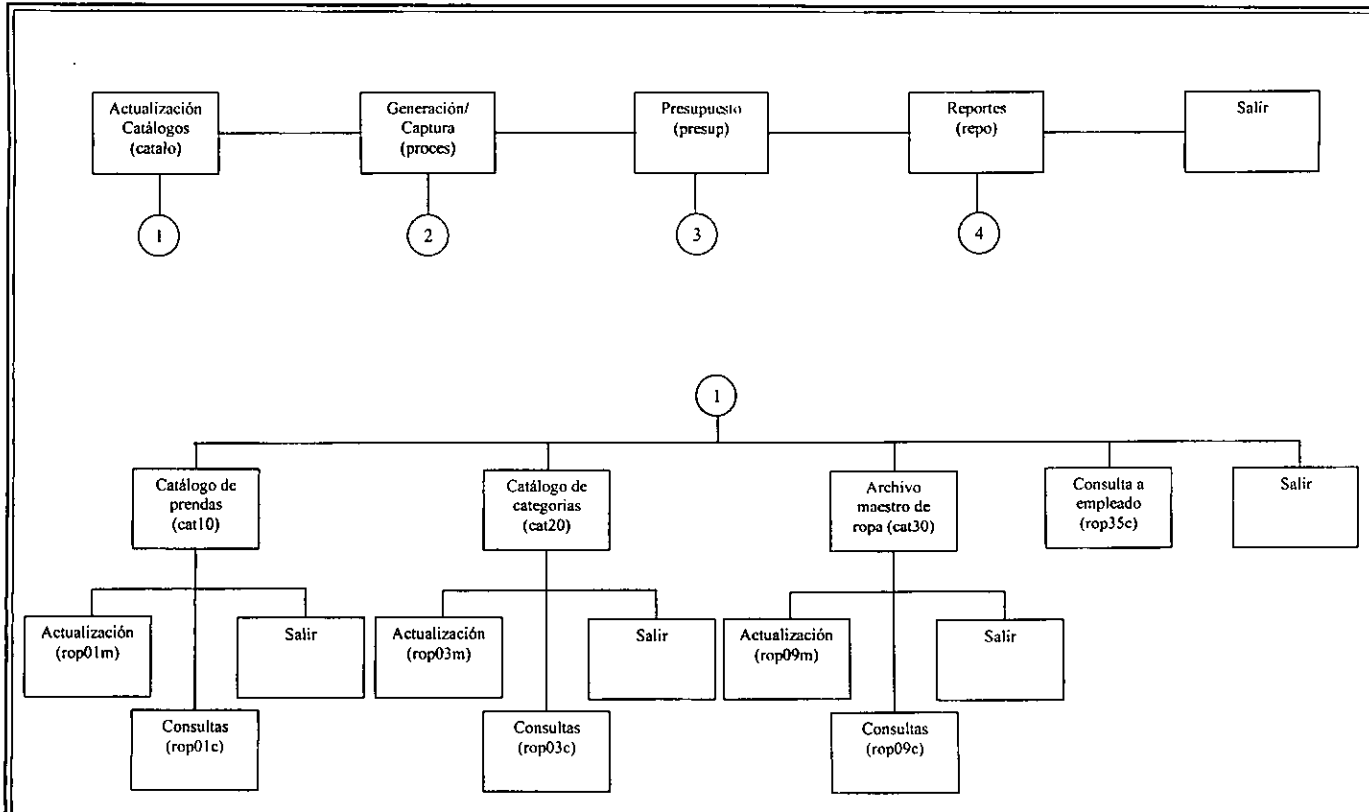


Figura 4.9 Gráfica de estructura

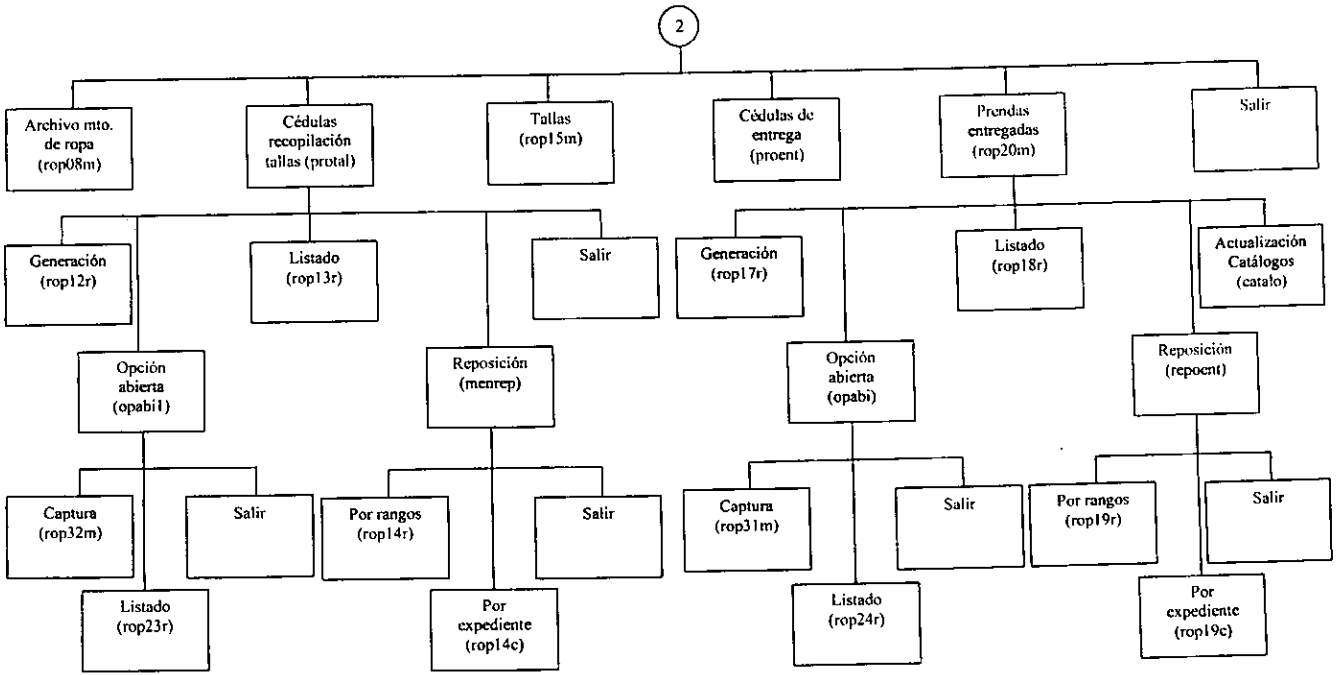


Figura 4.9 Gráfica de estructura (continuación)

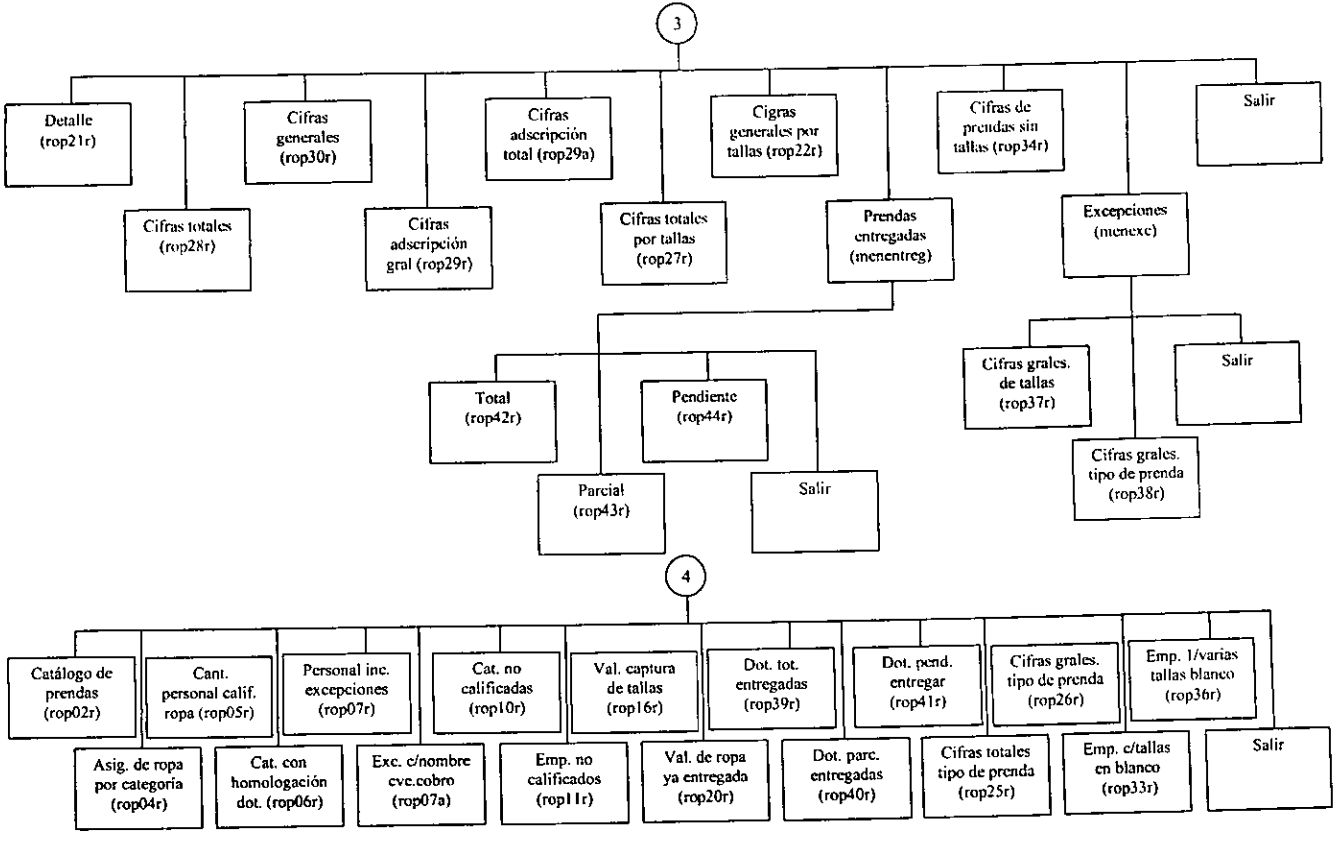


Figura 4.9 Gráfica de estructura (continuación)

4.2.2 DESARROLLO MODULAR

Un módulo que hace una llamada puede interactuar con más de un módulo subordinado.

Cuando un módulo llama a otro, el módulo que hace la llamada puede enviar datos al módulo llamado de tal forma que pueda llevar a cabo la función descrita en su nombre. De la misma forma, el módulo llamado puede producir datos que se transfieran de nuevo al módulo que hizo la llamada.

Los parámetros son datos necesarios en el módulo llamado para realizar el trabajo necesario. También se transfiere la información de control (bandera). Su propósito es ayudar a controlar el proceso indicado en la ocurrencia de, digamos, errores o condiciones de fin de archivo.

En la tabla 4.10 se muestran los principios para un buen desarrollo modular.

PRINCIPIO	DESCRIPCION	OBJETIVOS
Modularidad y fragmentación	Diseño de un sistema como una jerarquía de módulos	Diseñar la estructura en forma descendente con módulos que realicen funciones específicas.
Acoplamiento	La fuerza de las relaciones entre módulos	Maximizar la independencia entre los módulos minimizando el acoplamiento (reducir el acoplamiento del módulo).
Cohesión (Integración)	La fuerza de las relaciones dentro de un módulo	Maximizar la cohesión; los elementos altamente relacionados deben estar en el mismo módulo.
Extensión de control	Número de módulos subordinados al módulo que hace la llamada	Limitar la extensión de control de 5 a 7 módulos.
Tamaño	Número de instrucciones que componen a un módulo	Limitar el tamaño de forma que la función de todo el módulo se centre en un sólo propósito.
Uso compartido	Uso de un módulo por otros módulos	Evitar la duplicación permitiendo que los módulos sean llamados por otros que necesitan la función de cada uno.

Tabla 4.10 Principios para un buen desarrollo modular

Durante el estudio del diseño de entrada y menú, se enfatizó el enfoque descendente. El menú principal contiene varias opciones. Al escoger una de ellas se obtiene otro menú en el cual se presentan al usuario opciones más detalladas. Esta estructura proporciona a los usuarios un método fácil de entender para usar el sistema y elegir las opciones. No tienen que hacer todas las decisiones al mismo tiempo, sino solo una a la vez.

Cada función que el sistema lleva a cabo se identifica primero y después se desarrolla con mayor detalle; los procedimientos y procesos se desarrollan uno a la vez, desde lo general hasta lo particular.

El sistema esta formado por varios módulos separados que se llaman uno a la vez, cuando el usuario indica la función particular que desean llevar a cabo. A su vez el módulo de nivel superior llama a uno o varios módulos de nivel inferior hasta que la función deseada se realice.

4.2.3 ACOPLAMIENTO Y COHESIVIDAD

Acoplamiento : Los módulos del sistema deben tener poca dependencia entre si. Se refiere a la fuerza de la relación entre módulos de un sistema. En general se busco desarrollar la estructura del sistema de tal forma que un módulo tenga poca dependencia de cualquier otro módulo.

Se alcanzó un acoplamiento holgado (el cual minimizó la dependencia entre los módulos) de las siguientes formas:

- Controlar el número de parámetros que se transfieren entre los módulos.
- Evitar la transferencia innecesaria de datos a los módulos que se llamen.
- Transferir datos cuando sea necesario.
- Mantener las relaciones superior/inferior entre los módulos que llaman y los que son llamados.
- Preferentemente transferir datos, que información de control.

Cohesión : Los módulos deben llevar a cabo sólo una función de procesamiento. El contenido del módulo fue diseñado para que lleve a cabo una función especifica y para que sea más fácilmente entendible.

Hay cuatro tipos generales de contenidos de módulos. Los cuales se presentan en la tabla 4.11.

	CONTENIDO DE UN MODULO	EXPLICACION
MEJOR	Contenido de un módulo determinado por la función desarrollada	Todas las actividades de un módulo tienen el mismo propósito, es decir, llevar a cabo una función específica.
	Contenido de un módulo determinado por los datos usados	Todos los elementos en un módulo se refieren a los mismos datos o archivos.
	Contenido de un módulo determinado por la lógica de procesamiento	Todos los pasos se realizan juntos o manejan las mismas funciones.
PEOR	Contenido no relacionado de un módulo	Los módulos se desarrollan por tamaño o número de instrucciones (es el resultado de la programación con reglas escritas).

Tabla 4.11 Tipos de contenidos de módulos

El tipo menos recomendable de agrupación del contenido de un módulo esta formado por pasos que no llevan a cabo una función completa o que lógicamente no van juntos.

Un caso extremo hubiera sido que las actividades de entrada, salida, cálculo y manejo de archivos se llevarán a cabo en un único módulo, pero para que esto no ocurriera fue necesario revisar las especificaciones y tener un claro entendimiento de cómo manejar dichas tareas.

El contenido de un módulo también se puede agrupar debido a que va junto lógicamente. Un módulo que maneja todas las operaciones de entrada o las de salida o uno que maneja actividades de procesamiento, independientemente de las necesidades de manejo de datos, usa el agrupamiento lógico.

Los elementos del módulo también se pueden relacionar por el tiempo en el que se ejecutan; es decir, lógicamente parecen ir juntos y se llevan a cabo al mismo tiempo. Un módulo que inicializa todas las variables y abre archivos esta lógicamente integrado. Se noto que este nivel es mejor que el primer tipo, ya que todos los elementos son ejecutables en un mismo marco de tiempo.

Los módulos que están lógicamente integrados son difíciles de modificar, puesto que el costo será compartido por cada tipo de actividad. Hasta el más sencillo de los cambios puede afectar todos los tipos de transacción.

Una mejor solución fue separar cada tipo de transacción en su propio módulo.

El contenido de un módulo también se pudo determinar por los datos que usa. Un módulo en el que el contenido se refiere a los mismos datos es preferible a uno que sea desarrollado solo bajo la base de la lógica del procedimiento. Por ejemplo, se diseño un módulo para que todas las operaciones sobre un conjunto de datos se llevan a cabo al mismo tiempo; se prepara un archivo para imprimirse en papel, a disco y también a pantalla.

Un módulo que lee la siguiente transacción y actualiza el archivo maestro, añadiendo, eliminando o cambiando registros, incluyendo la verificación de error necesaria para cada tipo de función, comparte un conjunto común de datos. Este tipo de cohesión se observa en la generación del archivo maestro de uniformes y ropa de trabajo y es mejor que los otros tipos presentados, pero no tan bueno como el agrupamiento funcional.

El agrupamiento o cohesión funcional permite una prueba más global del módulo. Si se necesitaran cambios posteriores, se puede determinar rápidamente como se estructura el módulo y como procesa los datos e interactua con los demás módulos en el sistema.

Se enfatizo constantemente en la confiabilidad y facilidad de mantenimiento en todo el desarrollo.

4.2.4 NORMALIZACION

La normalización es el proceso de simplificar la relación entre campos de un registro. Por medio de la normalización, un conjunto de datos en un registro se reemplaza por varios registros que son más simples y predecibles y, por lo tanto, más manejables. La normalización se llevó a cabo por cuatro razones:

- Estructurar los datos de forma que se pudieran representar las relaciones pertinentes entre ellos.
- Permitir la recuperación sencilla de los datos en respuesta a las solicitudes de consulta y reportes.
- Simplificar el mantenimiento de los datos actualizándolos, insertándolos y borrándolos.
- Reducir la necesidad de reestructurar o reorganizar los datos cuando surgieran nuevas aplicaciones.

4.2.4.1 PASOS DE LA NORMALIZACION

1. Descomponer todos los grupos de datos en registros bidimensionales.
2. Eliminar todas las relaciones en las que los datos no dependan completamente de la llave primaria del registro.
3. Eliminar todas las relaciones que contengan dependencias transitivas.

PRIMERA FORMA NORMAL

Una de las mejoras básicas que se pudo tener es que se diseño la estructura de un registro de manera que todos los registros de un archivo tengan la misma longitud. Los

registros de longitud variable crean problemas especiales ya que el sistema verifica siempre en donde se encuentran los extremos de un registro.

La primera forma normal se alcanzó cuando se quitaron todos los grupos de repetición, de forma que un registro tuviera longitud fija (figura 4.12).

Un grupo de repetición, es decir, la aparición repetida de un dato o grupo de datos dentro de un registro es en realidad otra relación. Por lo tanto, se quita el registro y se le considera como una parte del mismo o como una relación adicional. La parte del registro de los datos que se repite se le denomina grupo de repetición.

Catálogo de prendas

rp1_cverh	rp1_cveal	rp1_desc
100	00170201	BATA DE GAB. AZUL PLUM
1000	00170416	CAMISA 100% ALGODÓN
1001	00171268	CAMISA POL / ALGODÓN

Catálogo de categorías

rp2_tnomi	rp2_adscr	rp2_categ	rp2_sexo	rp2_cont	rp2_ident	rp2_cverh1	rp2_canti1	rp2_cverh2	rp2_canti2	rp2_cverh3	rp2_canti3	rp2_cverh4	rp2_canti4	rp2_cverh5	rp2_canti5	rp2_cverh6	rp2_canti6
1	100000000	10102	M	10	U	1001	6	1600	1	1601	1	1603	1	1610	1	1611	1
2	20400000	70412	F	10	U	2230	1	2231	1	2232	1	2233	6	2240	1	2241	1
2	20610100	70519	M	12	U	109	3	205	1	1000	6	1200	1	1201	1	1203	1

Continuación del catálogo de categorías...

rp2_cverh7	rp2_canti7	rp2_cverh8	rp2_canti8	rp2_cverh9	rp2_canti9	rp2_cverh10	rp2_canti10	rp2_cverh11	rp2_canti11	rp2_cverh12	rp2_canti12	rp2_cverh13	rp2_canti13	rp2_cverh14	rp2_canti14	rp2_cverh15	rp2_canti15
1613	1	1620	1	1621	1	1623	1										
2242	1	2250	1	2251	1	2252	1										
1210	1	1211	1	1213	1	1220	1	1221	1	1223	1						

Figura 4.12 Primera forma normal (Cada registro tiene una longitud fija y no contiene grupos de repetición)

SEGUNDA FORMA NORMAL

Se alcanzó cuando un registro estaba en la primera forma normal y cada campo dependía totalmente de la llave del registro (en el almacenamiento y recuperación). En otras palabras, se buscó la dependencia funcional: Un campo es funcionalmente dependiente si su valor esta asociado de manera única con un campo especifico.

Para alcanzar la segunda forma normal, cada campo del registro que no dependa de la llave primaria debe quitarse y utilizarse para formar una relación aparte (figura 4.13)

Catálogo de prendas

rp1_cverh	rp1_cveal	rp1_desc
100	00170201	BATA DE GAB. AZUL PLUM
1000	00170416	CAMISA 100% ALGODÓN
1001	00171268	CAMISA POL / ALGODÓN

Catálogo de categorías

rp2_tnomi	rp2_adscr	rp2_categ	rp2_sexo	rp2_cont	rp2_ident	rp2_cverh1	rp2_canti1	rp2_cverh2	rp2_canti2	rp2_cverh3	rp2_canti3	rp2_cverh4	rp2_canti4	rp2_cverh5	rp2_canti5	rp2_cverh6	rp2_canti6
1	100000000	10102	M	10	U	1001	6	1600	1	1601	1	1603	1	1610	1	1611	1
2	20400000	70412	F	10	U	2230	1	2231	1	2232	1	2233	6	2240	1	2241	1
2	20610100	70519	M	12	U	109	3	205	1	1000	6	1200	1	1201	1	1203	1

Continuación del catálogo de categorías...

rp2_cverh7	rp2_canti7	rp2_cverh8	rp2_canti8	rp2_cverh9	rp2_canti9	rp2_cverh10	rp2_canti10	rp2_cverh11	rp2_canti11	rp2_cverh12	rp2_canti12	rp2_cverh13	rp2_canti13	rp2_cverh14	rp2_canti14	rp2_cverh15	rp2_canti15
1613	1	1620	1	1621	1	1623	1										
2242	1	2250	1	2251	1	2252	1										
1210	1	1211	1	1213	1	1220	1	1221	1	1223	1						

Figura 4.13 Segunda forma normal

Archivo maestro de ropa

rp3_numexp	rp3_ejerc	rp3_tnomi	rp3_adscr	rp3_categ	rp3_sexo	rp3_feact	rp3_fecofi	rp3_refofi	rp3_stent	rp3_sttal	rp3_repent	rp3_verent	rp3_reptal	rp3_vertal	rp3_flagexc	rp3_motrep	rp3_motcam	rp3_cont	rp3_ident
12675	2000	1	10000000	10102	M	02/08/2000												10	U
08078	2000	2	20400000	70412	F	14/02/2000												10	U
10273	2000	2	20610100	70519	M	27/01/2000												12	U

Continuación archivo maestro de ropa...

rp3_cverh1	rp3_canti1	rp3_cveen1	rp3_talla1	rp3_canen1	rp3_cverh2	rp3_canti2	rp3_cveen2	rp3_talla2	rp3_canen2	rp3_cverh3	rp3_canti3	rp3_cveen3	rp3_talla3	rp3_canen3	rp3_cverh4	rp3_canti4	rp3_cveen4	rp3_talla4	rp3_canen4
1001	6				1600	1				1601	1				1603	1			
2230	1		36		2231	1		36		2232	1		36		2233	6		36	
109	3		26		205	1		40		1000	6		38		1200	1		37	

Continuación archivo maestro de ropa...

rp3_cverh5	rp3_canti5	rp3_cveen5	rp3_talla5	rp3_canen5	rp3_cverh6	rp3_canti6	rp3_cveen6	rp3_talla6	rp3_canen6	rp3_cverh7	rp3_canti7	rp3_cveen7	rp3_talla7	rp3_canen7	rp3_cverh8	rp3_canti8	rp3_cveen8	rp3_talla8	rp3_canen8
1610	1				1611	1				1613	1				1620	1			
2240	1		36		2241	1		36		2242	1		36		2250	1		36	
1201	1		30		1203	1				1210	1		37		1211	1		30	

Figura 4.13 Segunda forma normal (continuación)

Continuación archivo maestro de ropa...

rp3_cverh9	rp3_canti9	rp3_cveen9	rp3_talla9	rp3_canen9	rp3_cverh10	rp3_canti10	rp3_cveen10	rp3_talla10	rp3_canen10	rp3_cverh11	rp3_canti11	rp3_cveen11	rp3_talla11	rp3_canen11	rp3_cverh12	rp3_canti12	rp3_cveen12	rp3_talla12	rp3_canen12
1621	1				1623	1													
2251	1		36		2252	1		36											
1213	1				1220	1		37		1221	1		30		1223	1			

Continuación archivo maestro de ropa...

rp3_cverh13	rp3_canti13	rp3_cveen13	rp3_talla13	rp3_canen13	rp3_cverh14	rp3_canti14	rp3_cveen14	rp3_talla14	rp3_canen14	rp3_cverh15	rp3_canti15	rp3_cveen15	rp3_talla15	rp3_canen15

Figura 4.13 Segunda forma normal (continuación)

TERCERA FORMA NORMAL

La tercera forma normal se alcanzó cuando se quitaron las dependencias transitivas del diseño de registro. El caso general se muestra en la figura 4.14.

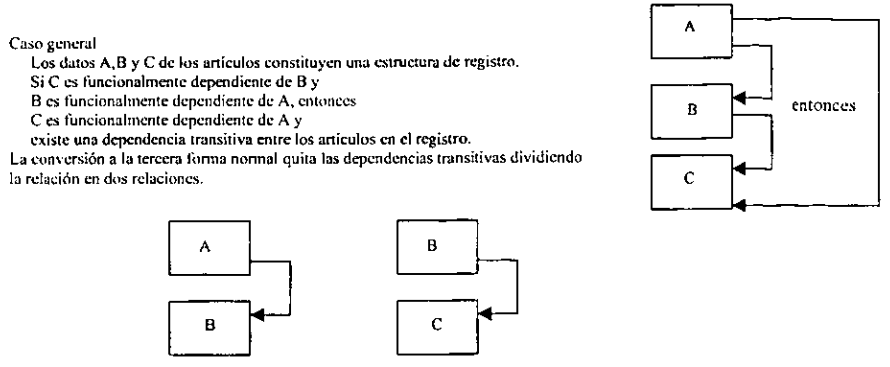


Figura 4.14 Tercera forma normal

- A, B y C son tres datos en un registro
- Si C es funcionalmente dependiente de B y
- B es funcionalmente dependiente de A
- Entonces C es funcionalmente dependiente de A
- Por lo tanto, existe una dependencia transitiva.

En el manejo de datos, la dependencia transitiva es una preocupación, ya que los datos pueden perderse de manera inadvertida cuando la relación esta oculta; en la figura 4.14, si se quita A, entonces también se quitan B y C, sea o no esta la intensión. Este problema se elimina diseñando el registro para la tercera forma normal. La conversión a la tercera forma normal quita la dependencia transitiva dividiendo la relación en dos relaciones separadas.

Los registros del sistema de uniformes y ropa de trabajo se hallan en la segunda forma normal. Pero puesto que no contienen dependencias transitivas, también están en la tercera forma normal (figura 4.13)

En la tabla 4.15 se resumen las tres formas de normalización analizadas. Y con esto se observó que si la base de datos se diseña de acuerdo con los principios de normalización, la manipulación de datos, será siempre mas fácil.

FORMA	PASOS
Primera forma normal	Cambiar todas las estructuras que no sean bidimensionales (es decir, grupos de repetición) en estructuras de registro bidimensionales.
Segunda forma normal	Eliminar los datos que no dependan totalmente de las llaves de registro.
Tercera forma normal	Eliminar los datos que dependan transitivamente de las llaves primarias.

Tabla 4.15 Formas de normalización

4.2.4.2 DIAGRAMA ENTIDAD-RELACION

Cuando se diseña un sistema de información para el procesamiento de transacciones, a menudo el centro de atención es una entidad.

Cuando el usuario adquiere experiencia con el sistema y surgen nuevos requerimientos de la aplicación, la atención cambia: de ser capaz de recuperar un registro específico a desarrollar la capacidad de relacionar los registros sobre distintas entidades. De esto precisamente se trata el manejo de datos: 1) marcar las relaciones naturales entre los datos y 2) compartir los datos entre entidades en todas las aplicaciones que necesiten de los detalles.

Por lo anterior, es útil mostrar las entidades y relaciones en forma gráfica por medio de los diagramas entidad-relación

Generalmente se representa una entidad por medio de un rectángulo con el nombre de la entidad dentro. El diamante indica la relación.

Las relaciones entre entidades se describen mediante su dependencia una de la otra, al igual que por el alcance de la relación.

DEPENDENCIA ENTRE ENTIDADES

Existen dos tipos de dependencia entre entidades. En la primera, la dependencia existencial, una entidad no puede existir a menos que la otra esté presente; el que exista la segunda depende de la existencia de la primera.

Al eliminar los registros de una entidad en una base de datos puede ocurrir que se eliminen los registros de otra si existe una dependencia existencial (esto no sucede en el sistema).

En el otro tipo de dependencia, la dependencia de identificación, una entidad no puede identificarse de manera única con sus propios atributos. La identificación es posible sólo mediante las relaciones de una entidad con otras. Para identificar una entidad, se deben conocer las otras.

ALCANCE DE LA DEPENDENCIA

El alcance de la dependencia incluye dos preocupaciones interrelacionadas: la dirección de la relación y el tipo de asociación entre ellas.

Una línea con flecha conecta a cada pareja de entidades. Dependiendo de la dirección de la flecha es como se lee la relación.

Las asociaciones entre entidades son uno a uno y uno a muchos y describen el alcance de la relación. Si es uno a uno, la aparición de una entidad quiere decir que existe una y sólo una aparición correspondiente de la otra entidad en la relación. Si es uno a muchos quiere decir, cero, una o más apariciones de la entidad correspondiente. El extremo final en forma de pata de gallina se usa típicamente para representar uno a muchos, al igual que la dirección de la relación.

Las relaciones indefinidas, en las que la dirección y asociación se desconocen, son inaceptables, ya que impiden el desarrollo de un modelo de datos que tenga sentido y por ende la parte de manejo de datos del sistema.

En la Figura 4.16 se muestra el diagrama entidad-relación del sistema.

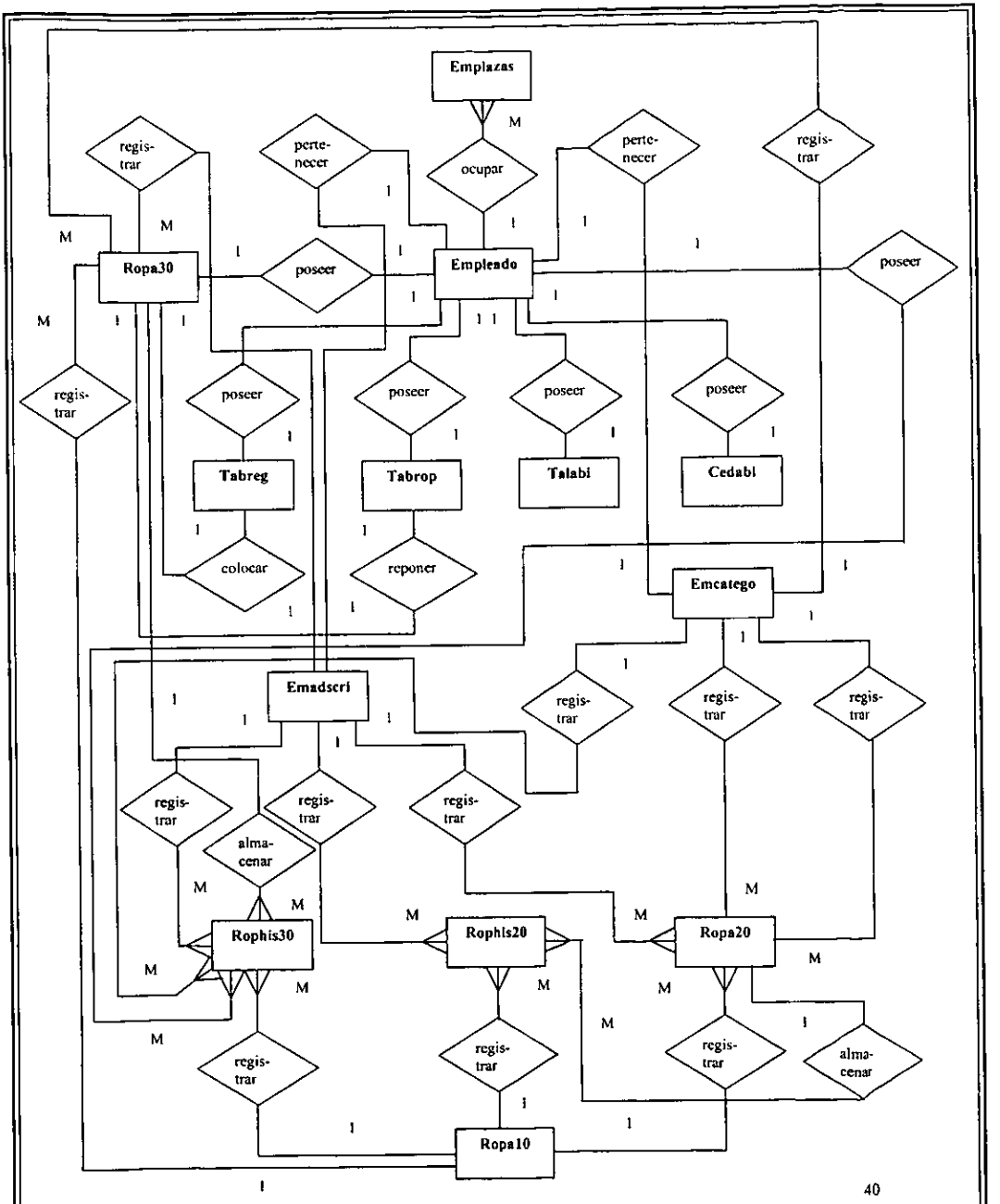


Figura 4.16 Diagrama Entidad-Relación

4.3 PROGRAMACION

Las especificaciones para programas son por sí mismas un diseño. Ellas describen cómo transformar las especificaciones de diseño - salidas, entradas, archivos, procesamiento y otras - en software.

El diseño del software fue importante para asegurar que:

- Los programas producidos llevarán a cabo todas las tareas y lo hicieran en forma establecida.
- La estructuración del software en módulos permitiera su prueba y validación para determinar si los procedimientos eran correctos.
- Las modificaciones futuras se podrían realizar en forma eficiente.

El sistema sería diseñado solo una vez, pero sería usado repetidamente y era muy probable que evolucionara en la medida que cambien las necesidades de los usuarios.

Por todo lo anterior se observó que era importante conocer el lenguaje de programación y la plataforma para el desarrollo de un buen sistema.

4.3.1. HERRAMIENTAS DE PROGRAMACION

EL SISTEMA OPERATIVO

El sistema UNIX (UNIX se deriva de MULTICS – Multiplexed Information and Computing System – y de UNICS – Uniplexed Information and Computing System – así UNICS dio lugar al nombre definitivo UNIX) proporciona un sistema operativo de tiempo compartido que controla las actividades y recursos de la computadora, y una interactiva interfaz operativa flexible. Fue diseñado para ejecutar múltiples procesos concurrentemente y soportar múltiples usuarios para facilitar la compartición de datos. El ambiente operativo fue diseñado con una arquitectura modular en todos los niveles. Los comandos que no saben nada acerca de otros se pueden combinar fácilmente por medio de entubamientos ("pipelines"), para realizar manipulaciones muy complejas.

La popularidad del sistema UNIX se puede atribuir enormemente a:

- La flexibilidad y que es muy completo, permitiéndole entrar en muchos ambientes de aplicación.
- Las numerosas utilerías incluidas en el ambiente operativo mejorando la productividad de los usuarios.
- La disponibilidad y portabilidad para muchas plataformas hardware.

En el sistema UNIX se pueden realizar varias tareas al mismo tiempo. Desde una simple terminal, el usuario puede ejecutar varios programas que parecen estar ejecutándose simultáneamente. Esto significa que un usuario puede editar un texto, mientras otro archivo sé esta formateando, no obstante que otro archivo sé este imprimiendo.

La capacidad multi-usuario permite que más de un usuario entre por medio de una cuenta (login) y use el sistema al mismo tiempo. Se pueden conectar múltiples terminales y teclados a la misma computadora. Si el sistema puede ejecutar simultáneamente múltiples programas, algunos de esos deben ser capaces de soportar otras sesiones de usuario. Adicionalmente, un usuario podrá entrar en varias ocasiones al mismo sistema a través de múltiples terminales. Una gran ventaja de esta arquitectura es que los miembros de un grupo de trabajo pueden tener acceso a los mismos datos al mismo tiempo, ya sea desde el punto de vista de desarrollo o de usuario.

LA LINEA DE PRODUCTOS INFORMIX

INFORMIX provee una línea de herramientas de software dirigidas al desarrollador de aplicaciones, para el manejo de la base de datos y para el desarrollo de aplicaciones en línea, así como una línea complementaria de productos dirigidos al usuario final, para soportar la toma de decisiones.

El servidor o manejador de base de datos utilizado es: INFORMIX OnLine, es rápido, tolerante a fallas que puede soportar grandes aplicaciones de operación intensiva, es totalmente relacional y esta basado en un lenguaje de consulta estructurada SQL (Structured Query Language) estándar de la industria.

ARQUITECTURA INFORMIX DE DOS PROCESOS: CLIENTE-SERVIDOR

INFORMIX utiliza una arquitectura de dos procesos (Cliente-Servidor) que separa el código de las aplicaciones de interfaz al usuario (front end), del código de las aplicaciones para el servidor de la base de datos (back end).

En una arquitectura de dos procesos, las herramientas y los servidores de base de datos pueden residir en la misma computadora o en diferentes computadoras conectadas vía una red. Cuando residen en diferentes computadoras, se denomina arquitectura Cliente-Servidor.

La aplicación cliente proporciona la interfaz necesaria para construir y ejecutar aplicaciones.

El manejador de la base de datos, o servidor, maneja toda la información de datos, incluyendo el almacenamiento y recuperación de los datos.

El SQL estándar se utiliza para comunicar las solicitudes de datos del cliente al servidor.

HERRAMIENTAS PARA EL DESARROLLO DE APLICACIONES

Las herramientas de INFORMIX para el desarrollo de aplicaciones en línea permiten al usuario crear aplicaciones que produzcan información útil en ambientes intensivos de transacciones. Estas herramientas son, básicamente INFORMIX-SQL y 4GL (4 Generation Language).

INFORMIX-SQL

Es un sistema para la administración de bases de datos relacionales basado en SQL. Proporciona las herramientas para construir aplicaciones de bases de datos:

- Facilidad de creación de menúes
- Editor de esquemas interactivo
- Lenguaje basado en el estándar SQL para consulta y definición
- Generador de formas
- Generador de reportes
- Utilería para la administración de la base de datos

INFORMIX-4GL

Es un lenguaje de programación de cuarta generación para bases de datos, que sigue el SQL estándar. Puede ser utilizado para crear aplicaciones complejas, totalmente adaptadas a los requerimientos del usuario y fáciles de modificar.

La versión de 4GL utilizada es: INFORMIX RAPID DEVELOPMENT SYSTEM que incorpora un procedimiento para la compilación y ejecución de programas 4GL, que eliminan la necesidad de un compilador "C", reduciendo significativamente el tiempo de compilación.

4.3.2 CARACTERISTICAS DEL LENGUAJE DE PROGRAMACION

La versión Rapid Development System usa las instrucciones de 4GL, una vez que se ha codificado un módulo, debe ser compilado y posiblemente ligado a otros módulos. El resultado de la compilación de INFORMIX-4GL RDS es "código p". El tiempo requerido para compilar el código p es muy corto. Esta es la razón por la que se llama "Rapid Development System", puesto que reduce dramáticamente el tiempo de compilación.

El código p generado se ejecuta a través de un programa "interprete" o runner, puesto que no es directamente ejecutable por la computadora. El programa interprete lee las instrucciones 4GL RDS codificadas y las traduce en tiempo de ejecución, a código directamente ejecutable por la máquina. INFORMIX-4GL RDS puede tomar un mayor tiempo para la ejecución. El tiempo dependerá del tipo de hardware y de lo que efectivamente haga el programa. Puede darse el caso, inclusive, de que no se observe ningún retraso.

LOS CONCEPTOS DE PROGRAMA, MODULO, FUNCION Y FORMA

Un programa en INFORMIX-4GL puede atender una amplia variedad de requerimientos del usuario. Para este propósito deberá de estar formado por uno o varios componentes. Sin embargo, el concepto tradicional de programa: una secuencia lógica de operaciones a ser ejecutadas por una computadora para la solución de un problema, se mantiene vigente.

En cualquier programa podemos observar operaciones básicas que cumplen funciones generales.

Uno de los tabiques constructivos de un programa es el módulo. Un módulo es un archivo de computadora con instrucciones de INFORMIX-4GL. Si se unen varios módulos se forma un programa. Cuántos módulos se necesitan para formar un programa?. Al menos uno. En general se requerirán tantos cuantos sean necesarios para cumplir los objetivos del programa.

Un módulo contiene una colección de una o más funciones. Las funciones son la unidad básica de programación de INFORMIX-4GL.

La función se utiliza para establecer un conjunto de instrucciones que deben de ser ejecutadas conjuntamente. Es posible que en un módulo haya funciones que se ejecuten en cada invocación del módulo, o bien que no se ejecuten nunca.

Cuántas funciones se requieren para construir un módulo?. Al menos una. Nuevamente, se requerirán tantas funciones cuantas sean necesarias para cumplir los objetivos del módulo. De hecho, puede haber varias otras funciones para manejar diferentes operaciones.

Una función es, en resumen, una rutina que efectúa un trabajo específico requerido en un programa. Existen cuatro tipos específicos de funciones en INFORMIX-4GL.

- **GLOBALS**

Este tipo de función contiene las instrucciones de declaración de las variables globales que serán utilizadas en los módulos del programa al que pertenece la función.

En cada programa sólo puede haber una función de tipo GLOBALS, en esta función sólo puede haber instrucciones de declaración.

- **MAIN**

Este tipo de función establece el punto de inicio de INFORMIX-4GL, este siempre leerá en primer lugar al módulo con la función MAIN, dicha función dirige al resto del programa. Por tanto, cada programa sólo puede tener una función MAIN.

- **REPORT**

Este tipo de función proporciona a 4GL las instrucciones adicionales para el formateo de un reporte. Otras funciones como MAIN, GLOBALS o FUNCTION no "entenderán" a las instrucciones de formateo.

- **FUNCTION**

Es la unidad básica de codificación de INFORMIX-4GL. No tiene un propósito específico como MAIN o GLOBALS. El programador define los propósitos de cada función.

Las entidades para interactuar mediante las pantallas son las FORMAS. La forma es un archivo separado que se usa para especificar el contenido del despliegue visual en la pantalla, ocurrido durante un proceso de entrada de datos, o un proceso de manipulación de información, o un proceso de consulta. Existe una gran flexibilidad en este proceso.

Las formas son entidades separadas del resto de un programa, no son propiamente un programa, ni un módulo, ni una función, sino que forman una categoría propia. Las formas son compiladas de manera separada y son usadas por un programa para desplegar información de una base de datos, así como para captar datos con los que sea actualizada.

EL AMBIENTE DE MENUES DE INFORMIX-4GL

Entrada al ambiente.

El acceso a la versión RDS es mediante la instrucción: r4gl.

El menú de 4gl.

Las opciones del primer menú presentado son:

Module	Conduce al menú (MODULE) para operar sobre un módulo.
Form	Conduce al menú (FORM) para operar sobre una forma.
Program	Permite especificar los componentes de un programa multimódulo.
Query-Language	Permite usar SQL de manera interactiva.
Exit	Permite regresar al sistema operativo.

Las opciones del menú (MODULE) son:

Modify	Permite hacer cambios a un módulo existente. Al seleccionar esta opción se requerirá escoger un módulo de una lista y 4GL conducirá a un editor. Al terminar los cambios y salir del editor, 4GL conducirá a los menúes de compilación.
New	Permite asignar un nombre para el módulo a compilar y conduce a un editor. Al terminar de escribir el módulo, 4GL conducirá a los menúes de compilación.
Compile	Permite escoger un módulo de una lista y conduce a los menúes de compilación.
Program_Compile	Permite compilar y ligar módulos, conforme a la especificación establecida en la base de datos. Es la misma opción que Compile en el menú PROGRAM.
Run	Presenta una lista de módulos compilados que pueden ser ejecutados.
Debug	Permite escoger un módulo ejecutable para ejecutar INFORMIX Interactive Debugger.
Exit	Regresa al menú INFORMIX-4GL.

Los menúes de compilación son utilizados para compilar el código 4GL, o bien como un módulo ejecutable, o bien como un módulo ligable. Las opciones de los menúes de compilación son:

Primer menú (MODIFY MODULE).

Compile	Permite ejecutar la compilación y conduce al segundo menú.
Save-and-exit	Guarda los cambios hechos durante la última sesión del editor y no intenta compilar .
Discard-and-exit	Elimina los cambios hechos durante la última sesión del editor.

Segundo menú (COMPILE MODULE).

Object	Permite compilar al módulo como parte de un programa multimódulos. Esto es, no puede correr por sí mismo.
Runable	Permite compilar al módulo como un programa, ejecutable por sí mismo. El módulo debe tener al menos MAIN y END MAIN.
Exit	Regresa al primer menú.

Tercer menú (COMPILE MODULE o bien MODIFY MODULE)

Son dos menúes alternativos, dependiendo de si el resultado de la compilación fue o no satisfactorio.

Menú COMPILE MODULE

Correct	Regresa al editor y señala los errores en el código. Una vez corregidos los errores regresa al primer menú.
Exit	Regresa al primer menú.

Menú MODIFY MODULE (Con las opciones ya vistas)

Menú (PROGRAM)

Este menú es utilizado para ligar los módulos de un programa.

Modify	Permite modificar los datos de la especificación del programa. Esta opción requiere el nombre del programa que se desea modificar. A continuación 4GL presenta una pantalla y un menú para actualizar la información del programa permite crear una nueva especificación con los módulos del programa y las bibliotecas que constituyen el programa requerido.
New	Permite crear una nueva especificación con los módulos del programa y las bibliotecas que constituyen al programa requerido.
Compile	Efectúa la compilación y liga del programa especificado, a partir de la fecha y hora de última actualización de los archivos. Coincide con la opción de Program-compile del menú MODULE.
Planned_Compile	Presenta un resumen de los pasos que serán ejecutados al seleccionar la opción de Compile, considerando la fecha y hora de la última actualización de los módulos que forman al programa.
Run	Presenta una lista de los programas compilados que pueden ser ejecutados.
Debug	Permite ejecutar al módulo seleccionado a través de Interactive Debugger.
Undefine	Elimina el programa seleccionado, pero no remueve a los módulos que lo forman.

La pantalla de datos asociada al menú (PROGRAM) aparecerá al seleccionar las opciones New o Modify. Los datos que despliega son: Program, Runner, Runner Path, Debugger Path, 4gl Source, 4gl Source Path, Global Source, Global Source Path, Other .4go, Other .4go Path.

Las opciones de menú que presenta son:

4GL	Permite actualizar la lista de fuentes 4GL. Aquí se captan los nombres de los módulos que forman a un programa y su trayectoria.
Globals	Permite actualizar el arreglo de fuentes globales. Si se efectúan cambios a este módulo, entonces todos los módulos serán recompilados, puesto que los cambios en este caso afectan a todos los módulos.
Other	Permite actualizar el arreglo de dos columnas nombradas Other .4go y Other .4go Path. Aquí se captan los nombres y ubicaciones de otros módulos objeto 4GL.
Program_Runner	Permite especificar el nombre del archivo y la trayectoria del interprete que debe correr al programa.
Rename	Permite renombrar al programa.
Exit	Regresa al menú PROGRAM.

CREACION DE UN PROGRAMA

Un programa es creado mediante la liga de una serie de módulos. El menú PROGRAM es un medio para registrar los módulos que forman a un programa.

Al escoger la opción New del menú PROGRAM por primera vez, 4GL preguntará si el usuario desea formar una base de datos de los fuentes 4GL. Esta base de datos se usará para guardar un registro de los módulos que forman a un programa. Se debe de contestar (Y) a esta pregunta. La respuesta permitirá crear una base de datos llamada syspgm4gl que registrará los módulos que forman al programa. Si la base de datos de los módulos fuente ya existe, entonces 4GL solicitará el nombre del nuevo programa. El nombre que escoja será usado para referirse a todos los módulos que forman al programa.

El nombre de un programa debe de comenzar con una letra, puede contener letras, números y guiones bajos (underscores) y puede ser de hasta 10 caracteres. Desde luego no puede ser una palabra reservada de INFORMIX-4GL.

En el apéndice C se muestra el código fuente de algunos módulos.

4.4 PRUEBAS

Esta fase fue sin duda una de las más importantes en el sentido de que a partir de ella se pudo corroborar si las fases anteriores se llevaron a cabo cumpliendo con los requerimientos establecidos.

También aquí se dio la pauta para pensar en la liberación exitosa con relación al tiempo estimado.

Sin embargo, no nos escapamos de ajustes que se hicieron necesarios en algunos de los módulos.

4.4.1 PARCIALES

En las pruebas parciales se probaron los programas que conforman al sistema, (Por esta razón a veces se llama a las pruebas parciales prueba de programas). Las unidades de software son los módulos y las rutinas que se ensamblan e integran para llevar a cabo una función específica. En este sistema se requieren varios módulos en varios niveles.

Las pruebas parciales se centraron primero en los módulos independientes entre sí, para localizar los errores. Esto permitió detectar errores en el código y lógica contenidos dentro de ese único módulo. Aquellos errores que resultarían de la interacción entre los módulos se evitan inicialmente. Para cada módulo se tiene la capacidad de introducir, cambiar o recuperar datos y responder a consultas o reportes.

Los casos de prueba necesarios para las pruebas parciales probaron cada condición u opción.

Si el módulo recibe una entrada y genera una salida, se necesitaron datos para examinar el rango de valores válidos e inválidos para ahí poder detectar rápidamente los errores existentes.

En varios casos las cantidades a imprimir eran mayores a la máscara que se colocaba en el código, por lo cual la impresión era errónea, debido a esto, se realizaron las correcciones necesarias.

Hay módulos que llevan a cabo iteraciones con procesos específicos contenidos dentro de un ciclo, se ejecutó cada condición con cero iteraciones, con una iteración y con el número máximo aproximado de iteraciones en el ciclo. Se examinaron en cada caso los resultados de la prueba ya que se notó la importancia de prestar especial atención a estas condiciones. Algunos procesos marcaron error al hacer cero iteraciones motivo por el cual se realizaron cambios en el código.

Las pruebas parciales se realizaron inicialmente en los módulos de nivel inferior y continuando de uno en uno. Para esto se proporcionaron datos necesarios, de esta forma se notó que el módulo se desempeñara en la forma en que lo haría al encajarse dentro

del sistema. Posteriormente se realizó la prueba ya con el módulo que integra todos los demás módulos, es decir, una prueba general.

4.4.2 DE ALMACENAMIENTO

Se determinó la capacidad del sistema para almacenar datos de transacciones, ésta se mide en términos del número de registros que un disco puede manejar o que un archivo puede contener. Estas capacidades están ligadas al espacio en disco y al tamaño de los índices, claves de registro y demás, pero éstos también se probaron.

La prueba de almacenamiento requirió almacenar continuamente datos. Al comparar las capacidades ofrecidas y las reales se verificó, por un lado, la exactitud de la documentación y permitió al mismo tiempo dar un juicio acerca de que capacidad real era óptima. Se notó que existe la capacidad suficiente de almacenamiento para las transacciones y los registros del archivo maestro.

En la tabla 4.17 se muestra el número de registros por cada archivo utilizado:

ARCHIVO	DESCRIPCION	NUMERO DE REGISTROS
CEDABI	Opción abierta de entrega de prendas	0 -10
EMADSCRI	Catálogo de adscripciones	129
EMCATEGO	Catálogo de categorías	198
EMPLAZAS	Maestro de plazas	13,283
EMPLEADO	Maestro de empleados	19,991
ROPHIS20	Histórico del mto. de categ.	58
ROPHIS30	Histórico del mto. U. y R. T.	279
ROPA10	Catálogo de prendas	358
ROPA20	Catálogo de categorías ropa	2,485
ROPA30	Maestro de U. y R. T.	12,983
TALABI	Opción abierta de cédulas de tallas	0 - 10
TABREG	Respaldo de tallas	12,965
TABROP	Reposiciones	0 - 10

Figura 4.17 Registros por archivo

El número de registros es el más aproximado, pero puede variar dependiendo de los movimientos realizados por los usuarios.

4.4.3 DE TIEMPO DE EJECUCION

Esta prueba se llevó a cabo antes de la implantación para determinar cuanto tiempo se lleva recibir una respuesta a una consulta, hacer una copia de respaldo de un archivo o mandar un proceso.

También se incluyeron corridas de prueba para conocer el tiempo necesario para indexar o reordenar grandes archivos del tamaño de los que tendría el sistema durante una corrida típica o bien preparar un reporte.

Antes de la implantación se pudieron hacer con mayor facilidad los ajustes, ya que con sólo algunos registros los procesos de prueba corrían bien, pero cabia la posibilidad de que se alentara al estar cargados los datos completos. Al cargar los archivos en su totalidad los cambios realizados fueron mínimos.

4.4.4 DE PROCEDIMIENTO

En esta etapa se probaron todos los procesos que constituían el sistema y sirvió como ayuda para la realización del manual de usuario y técnico, ya que el primero indica al usuario como llevar a cabo ciertas funciones. Al realizar esta prueba se incluyeron en dichos manuales descripciones de los detalles que se encontraban. Este tipo de prueba no sólo muestra donde se necesitaron, sino también en que lugar había errores, es decir, donde las acciones sugeridas en la documentación no eran compatibles con las que realmente había que llevar a cabo para hacer que el sistema funcione.

4.4.5 DEL SISTEMA

La prueba del sistema no probó el software en si, sino la integración de cada módulo. También se buscaron las discrepancias entre el sistema y su objetivo original, especificaciones y documentación. La preocupación principal fue la compatibilidad de los módulos individuales.

Se hallaron las áreas en donde los módulos habían sido diseñados con especificaciones distintas para la longitud y tipo de datos y los nombres de los elementos de los datos. Por ejemplo, un módulo esperaba que el tipo de dato fuera numérico, mientras otro esperaba que fuera tipo carácter. El sistema no mandaba error alguno, pero se previno esto para evitar que se dieran resultados inesperados.

En esta prueba también se verificaron que los tamaños de los archivos fueran los adecuados y que los índices se construyeran en forma correcta. Se probaron los procedimientos de ordenamiento y reindexación para ver que existían y que lograban los resultados que esperaban los módulos.

En todas las etapas de pruebas anteriores se utilizaron datos reales y artificiales.

Los datos reales se extrajeron de los archivos de la empresa, aunque fue difícil obtenerlos en cantidad suficiente para obtener una prueba extensa, pero se logró tener lo suficiente para tener un resultado satisfactorio.

Los datos artificiales se crearon solamente con fines de prueba, con ellos se pudieron probar todas las combinaciones de formatos y valores.

4.5 DOCUMENTACION

Una vez que se dio visto bueno, la documentación no se hizo esperar, para ello se definieron prototipos, tanto para el manual de usuario como el manual técnico.

La examinación sistemática de manuales dará una imagen de la forma en que el sistema trabaja.

4.5.1 MANUAL DE USUARIO

El manual de usuario contiene:

- Una descripción general de cada módulo, así como el objetivo del mismo y una lista de conceptos básicos.
- Una explicación detallada de cada una de las opciones que componen el módulo, incluyendo pantallas.
- Un anexo de reportes o documentos fuentes que sirven de complemento para el buen entendimiento de la operación.

4.5.1 MANUAL TECNICO

El manual técnico contiene:

- Una lista general de los programas que componen el módulo, así como una descripción del objetivo de cada uno de ellos.
- El diagrama jerárquico que esquematiza el módulo completo.
- Simbología utilizada en los diagramas de flujo
- Diagrama de flujo para cada programa

- Por cada programa una cédula descriptiva que refleja: Objetivo del mismo; Descripción de lo que realiza, resaltando puntos de interés o críticos; Archivo de variables globales si ocupa; Nombre de las formas que utiliza el programa según sea el caso; y desde luego las tablas de la base de datos que son utilizadas durante el proceso indicando el tipo de acceso (lectura o actualización).

- Un listado del código del programa.

- En el caso en que el programa genere un reporte, se anexa un ejemplo del mismo.

4.6 IMPLANTACION, MANTENIMIENTO Y EVALUACION DEL SISTEMA

El proceso de primero asegurarse de que el sistema fuera operacional y permitir que después tomaran los usuarios control de la operación para su uso y evaluación es llamado implantación.

La creación e implantación se propuso como una forma para facilitar a los usuarios a que satisfagan sus necesidades de información a corto plazo y que al mismo tiempo, todavía recibieran mantenimiento de parte del centro de computo. La evaluación fue para dar continuidad a la implantación.

4.6.1 INSTALACION

Para la instalación sólo se tuvieron que transferir el programa y las formas ejecutables; de igual manera un archivo de ayuda llamado por algunos módulos.

Se crearon los esquemas de las tablas necesarias para el funcionamiento del sistema, sobre la base de la herramienta de INFORMIX que permite dicha tarea.

4.6.2 CARGA DE ARCHIVOS

El control elemental consistió en asegurarse que todos los registros fueran introducidos al sistema. Por tal motivo se describe a continuación la forma en que se cargaron los archivos involucrados:

Con respecto a EMADSCRI (Catálogo de adscripciones), EMCATEGO (Catálogo de categorías), EMPLAZAS (Archivo maestro de plazas) y EMPLEADO (Archivo maestro de empleados) los datos ya estaban cargados puesto que son utilizados por otros sistemas.

Para los archivos ROPA10 (Catálogo de prendas) y ROPA20 (Catálogo de categorías), los datos tuvieron que ser introducidos en forma manual puesto que de éstos se obtendrían los datos que constituirían el archivo ROPA30 (Maestro de ropa).

Los archivos CEDABI (opción abierta de cédulas de entrega) y TALABI (Opción abierta de cédulas de tallas) se cargarían en base a la captura hecha por el usuario cada vez que así lo requiriera.

Al realizar algún cambio en el archivo de categorías de ropa o en el maestro de ropa, éste se grabaría en su respectivo archivo histórico ROPHIS20 y ROPHIS30.

El Archivo TABREG (Respaldo de tallas) se cargará cada vez que se genere el archivo maestro de ropa (ROPA30) ya que en éste se respaldan las tallas que se tienen en ese momento para posteriormente volver a colocarlas si es que el empleado no cambia de categoría.

TABROP (Reposiciones) se carga en base a la captura del usuario puesto que es en base a reposiciones que desean hacer.

Se entregaron listados de la actualización hecha manualmente para su validación confirmando así que la información era la correcta.

4.6.3 IDENTIFICACION DE RESULTADOS

Al realizar la implantación y tener ya datos en los archivos, se notó que los resultados eran los que se esperaban, la captura se realizó sin ningún problema, al igual que la generación del archivo maestro y de los reportes, ya que estos últimos, lo mismo que los procesos contenidos en el sistema cumplieron con los requerimientos hechos por el usuario. Se tuvieron que realizar algunos ajustes debido a que al ver físicamente el proceso, a pesar de los requerimientos, se decidió que sufrieran modificaciones motivo por el cual se da mantenimiento al sistema.

4.6.4 MANTENIMIENTO Y TIEMPO DE RESPUESTA

El mantenimiento implicó adaptaciones de versiones anteriores del software. Se llevó a cabo para adecuarse a los cambios en los reportes, archivos y datos o al haber necesidad de depurar y corregir errores o fallas en caso de emergencia.

La mayor cantidad de trabajo de mantenimiento fue para cumplir con peticiones de los usuarios, mejorar la documentación o recodificar los componentes del sistema para una mayor eficiencia.

Al estar en esta etapa del sistema se notó que las claves para reducir la necesidad de mantenimiento, al igual que para hacer posible que se realicen las tareas esenciales más eficientemente., son:

- Definir con mayor precisión los requerimientos del usuario durante el desarrollo.
- Preparar lo mejor posible la documentación.
- Usar métodos más efectivos para el diseño de la lógica del procedimiento.
- Hacer un mejor uso de las herramientas y técnicas existentes.
- Dirigir el proceso en forma efectiva.

Como lo indican los comentarios anteriores, el diseño es tanto un proceso como un producto. Se observó que las prácticas de diseño que se siguieron para el software afectaban en forma dramática la facilidad de mantenimiento. Las prácticas de diseño adecuadas dan por resultado un producto al cual puede dársele mantenimiento.

En la tabla 4.18 se resumen las clases de mantenimiento encontradas en el ambiente del sistema.

CATEGORIA	ACTIVIDAD
CORRECTIVO	Ajustes de emergencia, depuración rutinaria
ADAPTATIVO	Inclusión de cambios a los datos y archivos, así como al hardware y software.
PERFECTIVO	Mejoras solicitadas por los usuarios, mejoras en la documentación, recodificación para mejorar la eficiencia.

Figura 4.18 Tipos de mantenimiento del sistema

Los datos tecleados en la pantalla son guardados y no son transmitidos hasta que el usuario oprime una tecla o varias teclas indicadas como teclas de transmisión. Cuando se oprime alguna de estas teclas, los datos de la pantalla junto con cualquiera de las teclas que hayan sido oprimidas, son transmitidos. Mediante el uso de estas teclas, no se tiene que responder a cada teclazo de la terminal y por lo tanto se beneficia cada usuario del tiempo de respuesta mejorado.

Por lo tanto podemos decir que el tiempo de respuesta es el tiempo que transcurre entre el momento en que el usuario oprime una tecla y la aparición de una nueva pantalla generada por la computadora.

Dentro del sistema el usuario tiene la tecla ESC para aceptar un proceso y la tecla PAUSE para cancelarlo.

4.6.5 REVISIONES GENERALES

Después de implantar y completar los archivos, se hizo una revisión del sistema conducida de igual forma por los usuarios y el analista. Esto fue un proceso formal para determinar que tan bien estaba funcionando, como había sido aceptado y cuales ajustes eran necesarios.

La revisión fue importante para recabar información para el mantenimiento, ya que el sistema permanecería mientras no se requieran cambios debido a desarrollos internos, como nuevos usuarios o actividades de la empresa. La revisión después de la implantación fue la primera fuente de información de los requisitos de mantenimiento, y el interés fundamental fue determinar si el sistema cumplió su objetivo y se notó que el nivel de desempeño de los usuarios mejoró y que el sistema estaba produciendo el resultado deseado.

A continuación se mencionan algunas revisiones hechas a lo largo del proyecto:

REVISION DE REQUERIMIENTOS:

Fue un recorrido llevado a cabo para examinar las especificaciones de requerimientos formulados sobre la base del análisis. Se examinaron las funciones, actividades y procesos que el sistema debía manejar. Se enfatizó la información y las necesidades de procesamiento que tenía que manejar el diseño propuesto. Esto sirvió para poder enfrentar las inconsistencias entre las necesidades establecidas por los usuarios y las que se proponían cumplir.

Se incluyó documentación que previamente se leyó por las personas involucradas. Esta delineo sus características y necesidades e indicó la relación entre los componentes claves del sistema. Fue de vital importancia que se describieran y valuaran las fuentes y uso de la información.

REVISION DEL DISEÑO:

Como su nombre lo indica aquí se puso atención a las especificaciones del diseño para cumplir con requerimientos identificados previamente.

Su propósito fue determinar si el diseño propuesto cumplía con las necesidades efectiva y eficientemente. Si se hallaban discrepancias entre el diseño y los requerimientos se señalaron y estudiaron. El propósito no fue rediseñar, sino ver que se cumpliera realmente con lo solicitado por el usuario.

REVISION DE CODIGO:

Es un recorrido estructurado llevado a cabo con el fin de examinar el código de un programa desarrollado, junto con su documentación. Esto se trabajo con módulos individuales y componentes principales del programa. El código en si se comparó con las especificaciones originales para determinar si se satisfacían. Se realizó pensando en que podía pasar que una porción de código no coincidiera con lo establecido y mejor modificar

algún programa durante la revisión y no ya que se implantara; esto traería como resultado que los cambios fueran más fáciles y que el usuario recibiera el sistema adecuado.

En la revisión de programas también se vio la eficiencia de la ejecución, el uso de nombres estándares de datos y módulos y errores del programa.

REVISION DE PRUEBAS:

Se desarrollaron datos de prueba para detectar errores de diseño o del software.

El propósito de esta revisión fue hallar errores e involucrar al usuario para tener nuevas ideas y preguntas que se convertían en casos de prueba.

4.6.6 LA CALIDAD Y CONFIABILIDAD DEL SISTEMA

La revisión de los productos y documentación relacionada con el software para verificar su cobertura, corrección, confiabilidad y facilidad de mantenimiento aseguro la calidad. Y, por supuesto, incluyó la garantía de que el sistema cumplió con las especificaciones y los requerimientos para su uso y desempeño deseados.

Se analizaron cuatro niveles de aseguramiento de la calidad, los cuales se mencionan a continuación:

1.- PRUEBA:

Fue el proceso de ejecutar un programa con la intensión explícita de hallar errores, es decir, hacer que falle. Así, una prueba exitosa fue aquella que encontró un error.

Con esto se pudo asegurar más que el sistema tenía calidad, ya que al hacer fallar los programas se encontraban y solucionaban fallas que pudiesen tener ya en la implantación.

2.- VERIFICACION Y 3.- VALIDACION:

La verificación tuvo intensión de hallar errores. Se llevó a cabo ejecutando un programa en un ambiente simulado. La validación se refiere al proceso del uso del software en un ambiente no simulado para hallar sus errores. La retroalimentación de la fase de validación produjo cambios en el software para resolver los errores y fallas descubiertas. Se eligió un conjunto de instalaciones usuarias que pusieron a trabajar al sistema en un ambiente real. Estas instalaciones usaron el sistema en las actividades cotidianas, procesaron transacciones en directo y produjeron salidas normales. El sistema estuvo a prueba en toda la extensión de la palabra, excepto que los usuarios estaban inadvertidos que estaban usando un sistema que podía fallar. Sin embargo las transacciones que se procesaron y las personas que lo usaron fueron reales.

En el curso de la validación, ocurrieron fallas y el software se modificó.

4.- CERTIFICACION:

Es una garantía de lo correcto de un programa, es decir, que el software realmente hace lo que se dice y de manera apropiada cubría con los requerimientos establecidos.

La confiabilidad del sistema significó que los datos son confiables, que son precisos y creíbles. Por esto se tiene una contraseña que sólo la conoce el usuario y esta sirve para generar el archivo maestro de ropa que es el que contiene la información más importante existente. Con la instrucción propia del lenguaje de programación se pudo hacer una rutina con la cual se da acceso a la generación de dicho archivo, al teclear la palabra clave no se ven las letras en pantalla, sólo se mueve el cursor.

4.7 CAPACITACION

Tuvo indicios desde las pruebas y culminó con la impartición de cursos formales apoyados en los manuales de usuario y técnico.

La capacitación fue importante, ya que recibida por la persona relacionada con el sistema podía ayudar u obstruir y hasta llegar a impedir la implantación exitosa. Como esto era lo que menos se quería, todos aquellos que estuvieran asociados con el sistema o afectados por el mismo debían conocer con detalle cuales eran sus papeles, como podían usar el sistema y que haría o que no haría este. Tanto los técnicos como los usuarios del sistema necesitaron capacitación.

4.7.1 OPERATIVA

Se capacitó al usuario sobre como operar el equipo. Preguntas que parecían triviales se hicieron notar, por ejemplo: cómo encender el equipo, cuándo se debe apagar sin el peligro de perder datos, etc., ya que éste no esta familiarizado con la aplicación.

Se incluyó la identificación de los problemas, determinando si el problema que surgía era causado por el equipo, por el software o algo hecho por el usuario al usar el sistema.

La mayor parte de la capacitación tiene que ver con la operación del sistema en si, actividades de manejo de datos recibieron la mayor atención, como fueron: la captura de datos (como guarda nuevas transacciones), la edición de datos (como modificar los datos grabados previamente), la formulación de consultas (como localizar registros específicos u obtener respuestas a preguntas) y el borrado de registros. El grueso del uso

del sistema implicó este conjunto de actividades, lo cual quiere decir que la mayor parte del tiempo de la capacitación se dedicó a esta área.

Un aspecto importante en la capacitación fue la familiarización con el sistema de procesamiento en sí (es decir, el equipo usado para la captura y procesamiento de datos, los procesa y produce los resultados).

Es importante mencionar que se notó que la debilidad de cualquier aspecto de la capacitación traería la posibilidad de llegar a situaciones embarazosas que producirían en el usuario frustración, errores o ambos. Una buena documentación, aunque esencial, no reemplaza la capacitación. No hay sustituto para la operación directa del sistema mientras que se aprende su uso

4.7.2 TECNICA

El sistema depende del personal del Centro de Computo, el cual es responsable de mantener al equipo funcionando, así como de proporcionar el servicio de apoyo necesario. Su capacitación debió asegurar que podía manejar todas las operaciones posibles, tanto rutinarias como extraordinarias.

Se incluyeron aspectos tan básicos como saber prender el equipo, usarlo, apagarlo y también un conocimiento de lo que es su operación y uso normales. También se vieron los desperfectos más comunes, como reconocerlos y que pasos llevar a cabo cuando ocurrieran. Se dio una lista de formas de resolver los problemas, pero para poder utilizarla el encargado debía identificar el posible problema y su solución.

La capacitación también necesitó la familiarización con los procedimientos de ejecución, lo cual implico trabajar a través de la sucesión de actividades necesarias para usar el sistema. Estos procedimientos permitieron a los encargados familiarizarse con las acciones que deben realizar y saber cuándo deben ocurrir dichas acciones. Además, supieron cuanto tiempo le llevara a las aplicaciones correr bajo condiciones normales. Esta información es importante, tanto para que los usuarios puedan planear su trabajo como para identificar los procesos que corren más rápido o más lento de lo esperado.

También se dio una explicación detallada del código de los programas y la función que tiene cada uno dentro del sistema.

4.8 LIBERACION

La liberación se realizó con la entrega oficial de la documentación del sistema y una vez concluida la capacitación operativa y técnica.

4.8.1 APROBACION FINAL

Mediante un documento oficial, el usuario dio a conocer la aceptación del sistema, cumpliendo con todos los requerimientos establecidos y dando inicio a trabajar con el ejercicio nuevo para la distribución y control de uniformes y ropa de trabajo en el Sistema de Transporte Colectivo (Metro).

CAPITULO 5. CONCLUSIONES

Mediante este trabajo note que el software de computadora se ha convertido en algo muy importante. Es la máquina que conduce a la toma de decisiones, sirve como base a la investigación moderna y a la resolución de problemas, es un transformador, modificando, produciendo, mostrando y transmitiendo la información.

Al detectarse en el Sistema de Transporte Colectivo (Metro) el problema que consistía en que había una deficiencia en la forma de cómo se entregaba la ropa adecuada al personal para sus actividades, se vio la necesidad de desarrollar un nuevo sistema el cual consistiría de varias opciones las cuales debían servir para la entrega rápida y oportuna de dichas prendas.

Los diversos análisis y herramientas estudiados, se utilizaron para dar solución al problema, quedando el usuario satisfecho con el trabajo realizado.

Finalmente se pudo cumplir con lo establecido, dándome la oportunidad de aplicar los conocimientos adquiridos en un problema real.

CAPITULO 6. BIBLIOGRAFIA

Ingeniería de software

Richard E. Fairley

Mc. Graw Hill

Ingeniería de software

un enfoque práctico

4ª. Edición

Roger S. Pressman

Mc. Graw Hill

Análisis y diseño de

sistemas de información

2ª. edición

James A. Senn

Mc. Graw Hill

Fundamentos del sistema operativo UNIX

Manual del estudiante

Servicios educacionales HP

Versión D

11/15/1993

Desarrollo de aplicaciones con INFORMIX 4GL

Material propiedad intelectual de CIBERTEC, S.A. de C.V.

INFORMIX 4GL

Application development language for the

UNIX operating system

Reference manual

Versión 4.0

March 1990

INFORMIX 4GL by example

Versión 4.1

July 1991

APENDICE A

SIMBOLOGIA

Terminal

Lectura y Actualización

Proceso

Tabla de la base de datos

Reporte

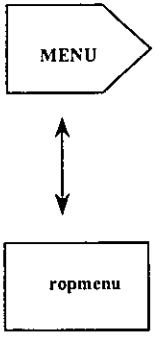
Sólo lectura de datos

Salida de información

Archivo Temporal

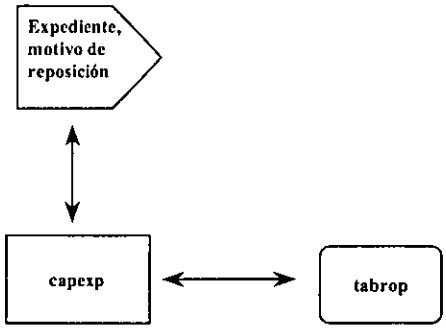
DESPLEGADO DE LAS OPCIONES DEL MENU PRINCIPAL

Módulo: ropmenu.4gl Menú principal.



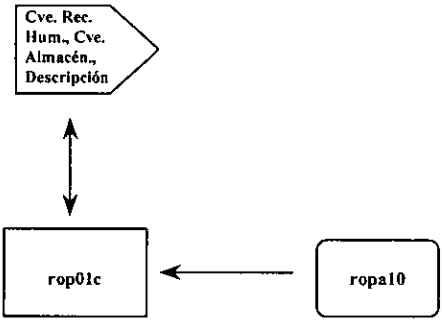
ALTAS EN ARCHIVO PARA REPOSICIONES

Módulo: capexp.4gl **Permite insertar registros para reposiciones.**



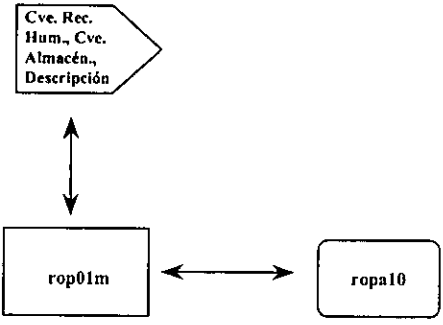
CONSULTAS POR CRITERIO AL CATALOGO DE PRENDAS

Módulo: rop01c.4gl Permite la consulta del catalogo de prendas.



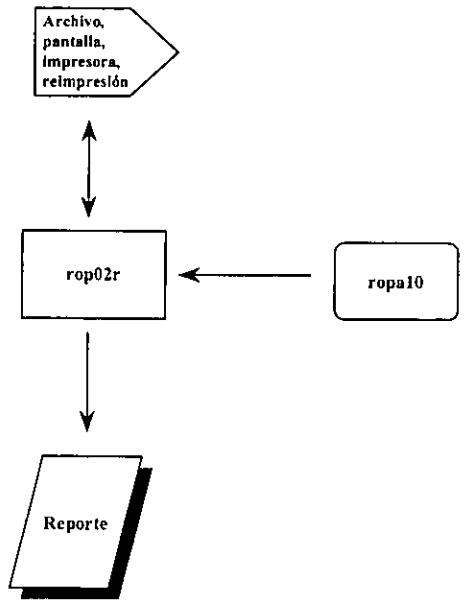
ACTUALIZACION AL CATALOGO DE PRENDAS

Módulo: rop01m.4gl Altas, bajas y cambios al catalogo de prendas.



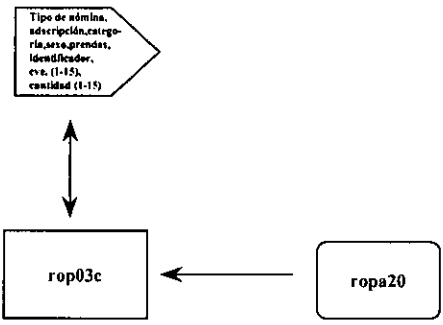
REPORTE DEL CATALOGO DE PRENDAS

Módulo: rop02r.4gl Genera reporte de la información contenida en el catalogo de prendas.



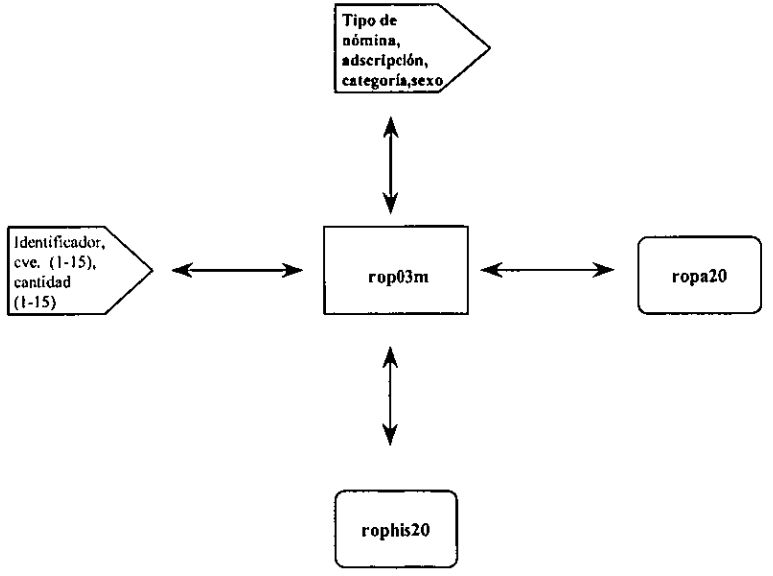
CONSULTAS POR CRITERIO AL CATALOGO DE CATEGORIAS

Módulo: rop03c.4gl Permite la consulta del catalogo de categorías.



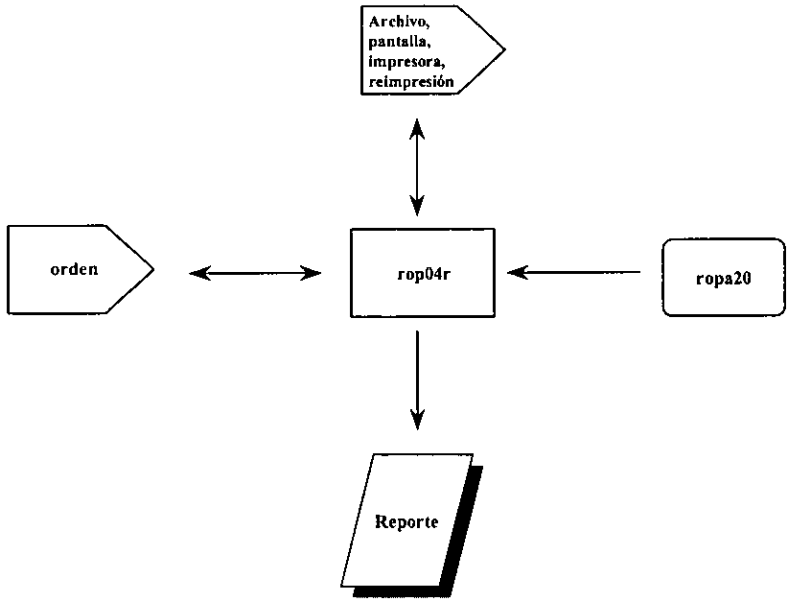
ACTUALIZACION AL CATALOGO DE CATEGORIAS

Módulo: rop03m.4gl Altas,bajas y cambios al catalogo de categorias.



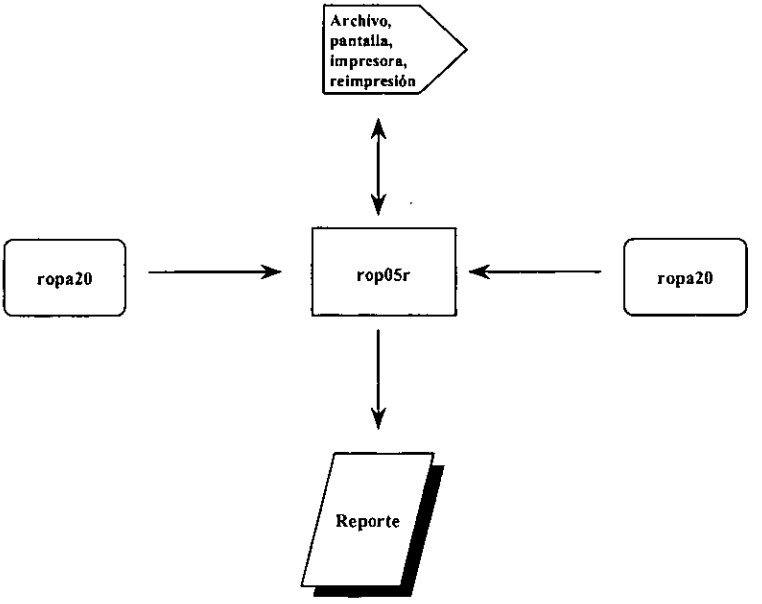
REPORTE DEL CATALOGO DE CATEGORIAS

Módulo: rop04r.4gl Genera reporte de la información contenida en el catalogo de categorias.



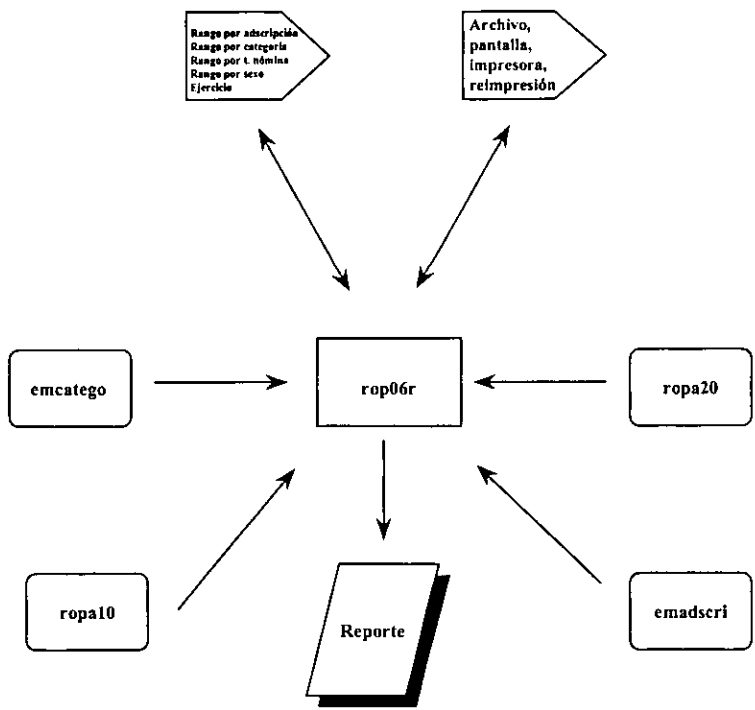
REPORTE DE TRABAJADORES POR CATEGORIA CALIFICADA

Módulo: rop05r.4gl Genera reporte de los empleados cuya categoría califique para asignación de uniforme o ropa de trabajo.



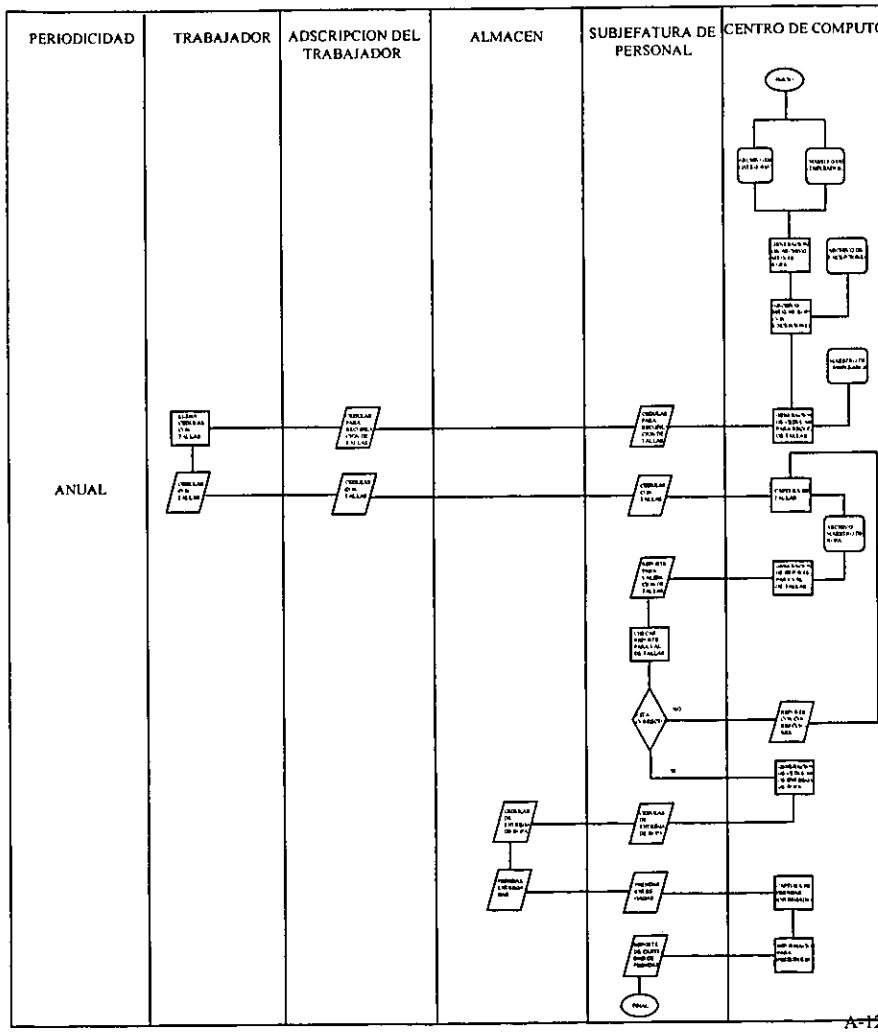
REPORTE DEL CATALOGO DE CATEGORIAS POR HOMOLOGACION DE DOTACIONES

Módulo: rop06r.4gl Genera reporte de las categorías cuya dotación de prendas es igual.



SISTEMA PARA LA DISTRIBUCION Y CONTROL DE UNIFORMES Y ROPA DE TRABAJO EN EL SISTEMA DE TRANSPORTE COLECTIVO (METRO)

DIAGRAMA GENERAL



APENDICE B

SISTEMA DE TRANSPORTE COLECTIVO METRO

Sistema de Uniformes y Ropa de Trabajo
Diccionario de Datos:

Base de Datos : "sirh"

TABLA : CEDABI	OPCION ABIERTA DE ENTREGA DE PRENDAS	2/2
	EXTENT SIZE	NEXT SIZE
MODULO : UNIFORMES Y ROPA DE TRABAJO	EXTENTS: 16	16

canti14	char(2),	Cantidad otorgada (14va. prenda).
talla14	char(5),	Talla del trabajador (14va. prenda).
cverh15	char(4),	Clave asignada en Recursos Humanos (15va. prenda).
canti15	char(2),	Cantidad otorgada (15va. prenda).
talla15	char(5),	Talla del trabajador (15va. prenda).

I N D I C E S

Sin Indices.

SISTEMA DE TRANSPORTE COLECTIVO METRO

Sistema de Uniformes y Ropa de Trabajo
Diccionario de Datos:

Base de Datos : "sirh"

TABLA : EMADSCRI	CATALOGO DE ADSCRIPCIONES		
		EXTENT SIZE	NEXT SIZE
MODULO : UNIFORMES Y ROPA DE TRABAJO	EXTENTS:	16	16

depart	char(5) not null,	Clave de adscripción (Código de conversión)
depstc	char(8) not null,	Clave de adscripción o departamento (original del STC)
descri	char(30),	Descripción de la adscripción
funcio	char(3) not null,	Clave de la función para presupuesto.
ramo	char(4),	Clave del ramo para presupuesto
sucurs	char(3),	Clave de la ruta de pago
cuenta	char(4),	Clave de la cuenta contable
scuenta	char(2)	Clave de la subcuenta contable

I N D I C E S

```
Create unique index "monica".ix423_1 on "monica".emadscri (depart);
```

SISTEMA DE TRANSPORTE COLECTIVO METRO

Sistema de Uniformes y Ropa de Trabajo
Diccionario de Datos:

Base de Datos : "sirh"

TABLA : EMCATEGO **CATALOGO DE CATEGORIAS**

		EXTENT SIZE	NEXT SIZE
MODULO : UNIFORMES Y ROPA DE TRABAJO	**EXTENTS:**	16	16

catstc	char(5) not null,	Clave de la categoría original.
descri	char(20),	Descripción de la categoría.
tarifa hora	decimal(11,2),	Importe de la tarifa por hora.
hora trab	decimal(4,2),	Número de horas a trabajar por jornada
cuota diaria	decimal(11,4),	Importe de la cuota diaria.
salario men	decimal(11,2),	Importe del salario mensual.
catego	char(4),	Clave de conversión de la categoría.
clase	char(1) not null,	A = Admtivo, T = Técnico, O = Operativo
nivel	char(1),	Número de nivel
aplica	char(1),	Tipo de nómina a la que se puede aplicar la categoría 1) Base, 2) Confianza 3) Eventual, 4) Confianza y Eventual
fechaa	date not null,	Fecha de alta de la categoría.
sobrev	char(1),	Identificador de una categorías sobre valuada.
fechac	date	Fecha de actualización de la categoría.

I N D I C E S

```
Create unique index "monica".241_22 on "monica".emcatego (catstc);
Create index "monica".catcon on "monica".emcatego (catego);
```

SISTEMA DE TRANSPORTE COLECTIVO METRO

Sistema de Uniformes y Ropa de Trabajo
Diccionario de Datos:

Base de Datos : "sirh"

TABLA : EMPLAZAS	MAESTRO DE PLAZAS	EXTENT SIZE	NEXT SIZE
MODULO : UNIFORMES Y ROPA DE TRABAJO	EXTENTS:	16	16

calstc	char(1) not null,	Clave de la calidad laboral 1)Base, 2) Confianza , 3) Eventual, 4) Base complementaria, 5) Mandos Medios, 6) Honorarios, 7) Mandos Superiores.
depart	char(5) not null,	Clave de la adscripción (conversión)
depstc	char(8) not null,	Clave de la adscripción (original STC)
catego	char(4) not null,	Clave de conversión de la categoría
catstc	char(5) not null,	Clave de la categoría original
plaza	char(4) not null,	Número de la plaza
cvepag	char(18) not null,	Clave de pago
tippla	char(1),	Clave del tipo de la plaza
sinemp	char(1),	Identificador del sindicato o empresa
secsin	char(2),	Identificador de la sección sindical
ocupor	char(10) not null,	Número de expediente del ocupante de la plaza
top	char(2) not null,	Identificador del tipo de ocupación de la plaza
propie	char(10) not null,	Número de expediente del propietario de la plaza
propue	char(2),	Identificador de la plaza propuesta
bloque	char(6),	Número de bloque en que la plaza es asignada
fechac	date,	Fecha de la última actualización
fechaa	date,	Fecha de alta de la plaza
tip_nom	char(1),	Identificador del tipo de nómina que pertenece a la plaza
usuari	char(10),	Clave del último usuario que actualizó
nobloq	char(4)	Número no completo del bloque en que se asigna la plaza
tipmov	char(2).	Clave del tipo de movimiento realizado
referencia	char(6).	Folio del documento de referencia
nopago	char(2).	Número del periodo en que se actualiza la plaza.

I N D I C E S

```
Create index "monica".ix274_2 on "monica".emplazas (depart);
Create index "monica".ix274_12 on "monica".emplazas (top);
Create index "monica".indtip on "monica".emplazas(tippla).
Create index "monica".inpat on "monica".emplazas(depstc, tip_nom).
Create index "monica".adp on "monica".emplazas(depstc).
Create index "monica".ctp on "monica".emplazas(catstc).
Create index "monica".tnp on "monica".emplazas(tip_nom).
```

TABLA : EMPLEADO MAESTRO DE EMPLEADOS 1/2

	EXTENT SIZE	NEXT SIZE
MODULO : UNIFORMES Y ROPA DE TRABAJO	EXTENTS: 16	16

numexp	char(10) not null,	Expediente
codigo_pago_frec	char(2),	Frecuencia de pago C) Catorcenal, Q) Quincenal, E) Especial, etc.
clase_empleado (1)	char(4),	Categoria
nombre_empleado	char(50),	Nombre del empleado
calle	char(30),	Calle
colonia	char(30),	Colonia
codigo_postal	char(10),	Código postal
fecha_nacimiento	integer,	Fecha Nacimiento
fec_nacimi	date,	Fecha de nacimiento formato date
cuenta_sar	char(30),	Cuenta del banco de (SAR)
desc_bancos	char(30),	Descripción del Banco (SAR)
cta_sar	char(18),	Cuenta del empleado (SAR)
rfc	char(16),	Registro Federal de Causantes
estado_civil	char(2),	Estado Civil
sexo	char(2),	"F" Femenino "M" Masculino
fecha contrato	integer,	Fecha de Ingreso STC
fec ingres	date,	Fecha de Ingreso al STC.formato date
fecha_antiguedad	integer,	Fecha de antigüedad en el sistema
fec_antigu	date,	Fecha de Antigüedad formato date
ult_aumento	integer,	Fecha de último aumento
fec_uttaum	date,	Fecha de último aumento formato date
fecha_baja	integer,	Fecha de baja
fec_baja	date,	Fecha de Baja en formato date
codigo_baja	char(2),	Clave de baja (Motivo de baja)
tipo de empleado	char(2),	Tipo de empleado 1) Base, 2) Confianza, 3) Eventual
flag_sindicato	char(2),	Identificador de cuota sindical
codigo_dept (2)	char(8),	Clave de departamento de la posición 1-5
sucursal	char(3),	Ruta de pago
fecrut	date,	Fecha de actualización de rutas de pago
tarifa_hora	decimal(11,2),	Tarifa salario por hora
horas_trab	decimal(4,2),	Horas trabajadas
cuota_diaria	decimal(11,4),	Cuota de salario diario
salario_men	decimal(11,2),	Salario mensual
zona_economica	char(4),	Zona económica, "9" Distrito Federal
forma_pago	char(2),	Formato de pago, "S" sobre
tipo_cuenta	char(2),	En caso de pagarse con cheque o depósito
desc_banco	char(10),	Descripción de campo
no_cta_banco	char(10),	Número de cuenta para depósito
flag_declara	char(2),	Identificador de declaración anual
pctje_imp_est	decimal(9,3),	Porcentaje de Impuesto Estatal
turno	char(2),	Clave de turno, 0) 8 horas, 1) 7 ½ hrs, 3) 7 hrs., 4) 6 ½ hrs.
callab	char(1) not null,	Clave calidad laboral, ej: O) Base, C/T) Confianza, E) Eventual,
calstc	char(1),	Clave calidad laboral, 1) Base, 2) Confianza, 3) Eventual, 4) Base complementaria, 5) Mando Medio, 6) Honorario, 7) Mando Superior
depart	char(5),	Clave de departamento (conversión)
depstc (2)	char(8) not null,	Clave de departamento (STC)
catego (1)	char(4) not null,	Clave de categoría (conversión)
catstc	char(5),	Clave de categoría (STC)

SISTEMA DE TRANSPORTE COLECTIVO METRO

Sistema de Uniformes y Ropa de Trabajo
Diccionario de Datos:

Base de Datos : "sirh"

TABLA : EMPLEADO	MAESTRO DE EMPLEADOS		2/2
		EXTENT SIZE	NEXT SIZE
MODULO : PERSONAL Y PLAZAS	EXTENTS:	16	16

plaza	char(4) not null,	Número de plaza del empleado
plazap	char(18),	Clave de plaza del empleado
cvepag	char(18),	Clave de pago (plaza que ocupa)
pzaant	char(18),	Clave de plaza anterior
fechaa	date not null,	Fecha de alta a la nomina
idbaja	char(1),	Identificador de baja "1" Baja
top	char(2) not null,	Tipo de ocupación de la plaza
tiplic	char(1),	Identificador de licencia, "0" Sin licencia, "1" Licencia sin goce de sueldo... (Ver catálogo de tipo de licencia)
fecter	date,	Fecha de término de contrato (Eventuales)
idmadr	char(1),	Identificador de ser madre
usuari	char(10) not null,	Identificador de usuario (actualizo)
idpena	char(1),	Identificador pensión alimenticia
totfal	smallint,	Total de faltas por empleado
totlic	smallint,	Total de licencias por empleado
idnomb	char(1) not null,	Identificador de nombramiento igual al top
fechac	date,	Fecha de la última actualización
idlact	char(1),	Identificador de lactancia
tipnom	char(1),	Identificador de tipo de nomina, 1) Base, 2) Confianza, 3) Eventual, 4) Base complementaria, 5) Mando Medio, 6) Honorario. 7) Mando superior
fechcc	date,	Fecha de cambio de categoría
fecint	date,	Fecha de término de interinato
fecrei	date,	Fecha de reingreso
motbaj	char(1),	Clave de motivo de baja
adscom	char(5),	Clave de departamento de comisión
feccom	date,	Fecha inicio de comisión
delega	char(2),	Código de delegación
entida	char(2),	Código de la entidad federativa
munici	char(2),	Código del municipio
cvelad	char(3),	Clave lada
telefo	char(7),	Teléfono del empleado
lugnac	char(2),	Lugar de nacimiento
clinic	char(1),	Clínica medica.
numpag	char(2)	Número de pago del periodo

I N D I C E S

Create unique index "monica".242_23 on "monica".empleado (numexp);
Create index "monica".lx651_4 on "monica".empleado (nombre_empleado);

Los campos marcados como 1) 2) son datos repetidos

SISTEMA DE TRANSPORTE COLECTIVO METRO

Sistema de Uniformes y Ropa de Trabajo
Diccionario de Datos:

Base de Datos : "sirh"

TABLA : ROPHIS20	HISTORICO DEL CATALOGO DE CATEGORIAS	EXTENT SIZE	NEXT SIZE
MODULO : UNIFORMES Y ROPA DE TRABAJO	EXTENTS:	16	16

rp2_tipnom	char(1),	Tipo de nómina 1) Base 2) Confianza 3) Eventual 4) Base Complementaria
rp2_adscr	char(8),	Clave de departamento (STC).
rp2_categ	char(5),	Clave de categoría (STC).
rp2_sexo	char(2),	Sexo F) Femenino M) Masculino.
rp2_fechis	datetime year to second,	Fecha y hora de actualización del registro.
rp2_cont	smallint,	Número de prendas correspondientes.
rp2_tipmov	char(1),	Tipo de movimiento A) Alta B) Baja C) Cambio.
rp2_ident	char(1),	Identificador de vestuario U) Uniforme R) Ropa A) Ambos
rp2_cverh1	char(4),	Clave asignada en Recursos Humanos (1ra. prenda).
rp2_canti1	char(2),	Cantidad otorgada (1ra. prenda).
rp2_cverh2	char(4),	Clave asignada en Recursos Humanos (2da. prenda).
rp2_canti2	char(2),	Cantidad otorgada (2da. prenda).
rp2_cverh3	char(4),	Clave asignada en Recursos Humanos (3ra. prenda).
rp2_canti3	char(2),	Cantidad otorgada (3ra. prenda).
rp2_cverh4	char(4),	Clave asignada en Recursos Humanos (4a. prenda).
rp2_canti4	char(2),	Cantidad otorgada (4a. prenda).
rp2_cverh5	char(4),	Clave asignada en Recursos Humanos (5a. prenda).
rp2_canti5	char(2),	Cantidad otorgada (5a. prenda).
rp2_cverh6	char(4),	Clave asignada en Recursos Humanos (6a. prenda).
rp2_canti6	char(2),	Cantidad otorgada (6a. prenda).
rp2_cverh7	char(4),	Clave asignada en Recursos Humanos (7a. prenda).
rp2_canti7	char(2),	Cantidad otorgada (7a. prenda).
rp2_cverh8	char(4),	Clave asignada en Recursos Humanos (8va. prenda).
rp2_canti8	char(2),	Cantidad otorgada (8va. prenda).
rp2_cverh9	char(4),	Clave asignada en Recursos Humanos (9a. prenda).
rp2_canti9	char(2),	Cantidad otorgada (9a. prenda).
rp2_cverh10	char(4),	Clave asignada en Recursos Humanos (10a. prenda).
rp2_canti10	char(2),	Cantidad otorgada (10a. prenda).
rp2_cverh11	char(4),	Clave asignada en Recursos Humanos (11va. prenda).
rp2_canti11	char(2),	Cantidad otorgada (11va. prenda).
rp2_cverh12	char(4),	Clave asignada en Recursos Humanos (12va. prenda).
rp2_canti12	char(2),	Cantidad otorgada (12va. prenda).
rp2_cverh13	char(4),	Clave asignada en Recursos Humanos (13va. prenda).
rp2_canti13	char(2),	Cantidad otorgada (13va. prenda).
rp2_cverh14	char(4),	Clave asignada en Recursos Humanos (14va. prenda).
rp2_canti14	char(2),	Cantidad otorgada (14va. prenda).
rp2_cverh15	char(4),	Clave asignada en Recursos Humanos (15va. prenda).
rp2_canti15	char(2),	Cantidad otorgada (15va. prenda).

I N D I C E S

Create unique index "monica".hisrop20 on "monica".rophis20 (rp2_tnomi, rp2_adscr, rp2_categ, rp2_sexo, rp2_fechis);

SISTEMA DE TRANSPORTE COLECTIVO METRO

Sistema de Uniformes y Ropa de Trabajo
Diccionario de Datos:

Base de Datos : "sirh"

TABLA : ROPHIS30	HISTORICO DEL MAESTRO DE UNIFORMES Y ROPA DE TRABAJO		1/2
		EXTENT SIZE	NEXT SIZE
MODULO : UNIFORMES Y ROPA DE TRABAJO	EXTENTS:	16	16

rp3_numexp	char(10),	Expediente.
rp3_ejerc	char(4),	Ejercicio activo.
rp3_tnomi	char(1),	Tipo de nómina 1) Base 2) Confianza 3) Eventual 4) Base Complementaria
rp3_adscr	char(8),	Clave de departamento (STC).
rp3_categ	char(5),	Clave de categoría (STC).
rp3_sexo	char(2),	Sexo F) Femenino M) Masculino.
rp3_feact	date,	Fecha de actualización dada por el usuario.
rp3_fechis	datetime year to second,	Fecha y hora de actualización del registro.
rp3_fecofi	date,	Fecha de oficio.
rp3_refofi	char(10),	Referencia de oficio.
rp3_stent	smallint,	Estatus de cédula de entrega. 1) Entregado.
rp3_sttal	smallint,	Estatus de cédula de tallas. 1) Entregado.
rp3_repent	smallint,	Cantidad de reposiciones de cédula de entrega.
rp3_verent	smallint,	Cantidad de versiones de cédula de entrega.
rp3_reptal	smallint,	Cantidad de reposiciones de cédula de tallas.
rp3_vertal	smallint,	Cantidad de versiones de cédula de tallas.
rp3_flagexc	char(1),	Identificador de excepción E) Excepción.
rp3_motrep	char(1),	Motivo de reposición.
rp3_motcam	char(1),	Motivo de cambio.
rp3_cont	smallint,	Número de prendas correspondientes.
rp3_ident	char(1),	Identificador de vestuario U) Uniforme R) Ropa A) Ambos.
rp3_tipmov	char(1),	Tipo de movimiento A) Alta B) Baja C) Cambio.
rp3_cverh1	char(4),	Clave asignada en Recursos Humanos (1ra. prenda).
rp3_canti1	char(2),	Cantidad otorgada (1ra. prenda).
rp3_talla1	char(5),	Talla del trabajador (1ra. prenda).
rp3_cverh2	char(4),	Clave asignada en Recursos Humanos (2da. prenda).
rp3_canti2	char(2),	Cantidad otorgada (2da. prenda).
rp3_talla2	char(5),	Talla del trabajador (2da. prenda).
rp3_cverh3	char(4),	Clave asignada en Recursos Humanos (3ra. prenda).
rp3_canti3	char(2),	Cantidad otorgada (3ra. prenda).
rp3_talla3	char(5),	Talla del trabajador (3ra. prenda).
rp3_cverh4	char(4),	Clave asignada en Recursos Humanos (4a. prenda).
rp3_canti4	char(2),	Cantidad otorgada (4a. prenda).
rp3_talla4	char(5),	Talla del trabajador (4a. prenda).
rp3_cverh5	char(4),	Clave asignada en Recursos Humanos (5a. prenda).
rp3_canti5	char(2),	Cantidad otorgada (5a. prenda).
rp3_talla5	char(5),	Talla del trabajador (5a. prenda).
rp3_cverh6	char(4),	Clave asignada en Recursos Humanos (6a. prenda).
rp3_canti6	char(2),	Cantidad otorgada (6a. prenda).
rp3_talla6	char(5),	Talla del trabajador (6a. prenda).
rp3_cverh7	char(4),	Clave asignada en Recursos Humanos (7a. prenda).
rp3_canti7	char(2),	Cantidad otorgada (7a. prenda).
rp3_talla7	char(5),	Talla del trabajador (7a. prenda).
rp3_cverh8	char(4),	Clave asignada en Recursos Humanos (8va. prenda).
rp3_canti8	char(2),	Cantidad otorgada (8va. prenda).

TABLA : ROPHIS30	HISTORICO DEL MAESTRO DE UNIFORMES Y ROPA DE TRABAJO	2/2
	EXTENT SIZE	NEXT SIZE
MODULO : UNIFORMES Y ROPA DE TRABAJO	EXTENTS: 16	16

rp3_talla8	char(5),	Talla del trabajador (8va. prenda).
rp3_cverh9	char(4),	Clave asignada en Recursos Humanos (9a. prenda).
rp3_canti9	char(2),	Cantidad otorgada (9a. prenda).
rp3_talla9	char(5),	Talla del trabajador (9a. prenda).
rp3_cverh10	char(4),	Clave asignada en Recursos Humanos (10a. prenda).
rp3_canti10	char(2),	Cantidad otorgada (10a. prenda).
rp3_talla10	char(5),	Talla del trabajador (10a. prenda).
rp3_cverh11	char(4),	Clave asignada en Recursos Humanos (11va. prenda).
rp3_canti11	char(2),	Cantidad otorgada (11va. prenda).
rp3_talla11	char(5),	Talla del trabajador (11va. prenda).
rp3_cverh12	char(4),	Clave asignada en Recursos Humanos (12va. prenda).
rp3_canti12	char(2),	Cantidad otorgada (12va. prenda).
rp3_talla12	char(5),	Talla del trabajador (12va. prenda).
rp3_cverh13	char(4),	Clave asignada en Recursos Humanos (13va. prenda).
rp3_canti13	char(2),	Cantidad otorgada (13va. prenda).
rp3_talla13	char(5),	Talla del trabajador (13va. prenda).
rp3_cverh14	char(4),	Clave asignada en Recursos Humanos (14va. prenda).
rp3_canti14	char(2),	Cantidad otorgada (14va. prenda).
rp3_talla14	char(5),	Talla del trabajador (14va. prenda).
rp3_cverh15	char(4),	Clave asignada en Recursos Humanos (15va. prenda).
rp3_canti15	char(2),	Cantidad otorgada (15va. prenda).
rp3_talla15	char(5),	Talla del trabajador (15va. prenda).

I N D I C E S

Create unique index "monica".rhis30 on "monica".rophis30 (rp3_numexp, rp3_ejerc, rp3_fechis);

SISTEMA DE TRANSPORTE COLECTIVO METRO

Sistema de Uniformes y Ropa de Trabajo
Diccionario de Datos:

Base de Datos : "sirh"

TABLA : ROPA10	CATALOGO DE PRENDAS	EXTENT SIZE	NEXT SIZE
MODULO : UNIFORMES Y ROPA DE TRABAJO	EXTENTS:	16	16

rp1_cverh	char(4) not null,	Clave asignada en Recursos Humanos.
rp1_cveal	char(8) not null,	Clave asignada en Almacén.
rp1_desc	char(30) not null,	Descripción de la prenda.

I N D I C E S

```
Create unique index "monica".rpa_cverh on "monica".ropa10 (rp1_cverh);
Create index "monica".rp1_cveal on "monica".ropa10 (rp1_cveal);
```

SISTEMA DE TRANSPORTE COLECTIVO METRO

Sistema de Uniformes y Ropa de Trabajo
Diccionario de Datos:

Base de Datos : "sirh"

TABLA : ROPA20	CATALOGO DE CATEGORIAS	EXTENT SIZE	NEXT SIZE
MODULO : UNIFORMES Y ROPA DE TRABAJO	EXTENTS:	16	16

rp2_tnomi	char(1),	Tipo de nómina 1) Base 2) Confianza 3) Eventual 4) Base Complementaria
rp2_adscr	char(8),	Clave de departamento (STC).
rp2_categ	char(5),	Clave de categoría (STC).
rp2_sexo	char(2),	Sexo F) Femenino M) Masculino.
rp2_cont	smallint,	Número de prendas correspondientes.
rp2_ident	char(1),	Identificador de vestuario U) Uniforme R) Ropa A) Ambos
rp2_cverh1	char(4),	Clave asignada en Recursos Humanos (1ra. prenda).
rp2_canti1	char(2),	Cantidad otorgada (1ra. prenda).
rp2_cverh2	char(4),	Clave asignada en Recursos Humanos (2da. prenda).
rp2_canti2	char(2),	Cantidad otorgada (2da. prenda).
rp2_cverh3	char(4),	Clave asignada en Recursos Humanos (3ra. prenda).
rp2_canti3	char(2),	Cantidad otorgada (3ra. prenda).
rp2_cverh4	char(4),	Clave asignada en Recursos Humanos (4a. prenda).
rp2_canti4	char(2),	Cantidad otorgada (4a. prenda).
rp2_cverh5	char(4),	Clave asignada en Recursos Humanos (5a. prenda).
rp2_canti5	char(2),	Cantidad otorgada (5a. prenda).
rp2_cverh6	char(4),	Clave asignada en Recursos Humanos (6a. prenda).
rp2_canti6	char(2),	Cantidad otorgada (6a. prenda).
rp2_cverh7	char(4),	Clave asignada en Recursos Humanos (7a. prenda).
rp2_canti7	char(2),	Cantidad otorgada (7a. prenda).
rp2_cverh8	char(4),	Clave asignada en Recursos Humanos (8va. prenda).
rp2_canti8	char(2),	Cantidad otorgada (8va. prenda).
rp2_cverh9	char(4),	Clave asignada en Recursos Humanos (9a. prenda).
rp2_canti9	char(2),	Cantidad otorgada (9a. prenda).
rp2_cverh10	char(4),	Clave asignada en Recursos Humanos (10a. prenda).
rp2_canti10	char(2),	Cantidad otorgada (10a. prenda).
rp2_cverh11	char(4),	Clave asignada en Recursos Humanos (11va. prenda).
rp2_canti11	char(2),	Cantidad otorgada (11va. prenda).
rp2_cverh12	char(4),	Clave asignada en Recursos Humanos (12va. prenda).
rp2_canti12	char(2),	Cantidad otorgada (12va. prenda).
rp2_cverh13	char(4),	Clave asignada en Recursos Humanos (13va. prenda).
rp2_canti13	char(2),	Cantidad otorgada (13va. prenda).
rp2_cverh14	char(4),	Clave asignada en Recursos Humanos (14va. prenda).
rp2_canti14	char(2),	Cantidad otorgada (14va. prenda).
rp2_cverh15	char(4),	Clave asignada en Recursos Humanos (15va. prenda).
rp2_canti15	char(2),	Cantidad otorgada (15va. prenda).

I N D I C E S

```
Create unique index "monica".rp2_tacs on "monica".ropa20 (rp2_tnomi, rp2_adscr, rp2_categ, rp2_sexo);
Create index  "monica".adin on "monica".ropa20 (rp2_tnomi) ;
```

TABLA : ROPA30 MAESTRO DE UNIFOMES Y ROPA DE TRABAJO 1/2

MODULO : UNIFORMES Y ROPA DE TRABAJO EXTENTS: EXTENT SIZE 16 NEXT SIZE 16

rp3_numexp	char(10) not null,	Expediente.
rp3_ejerc	char(4) not null,	Ejercicio activo.
rp3_tnomi	char(1) not null,	Tipo de nómina 1) Base 2) Confianza 3) Eventual 4) Base Complementaria
rp3_adscr	char(8) not null,	Clave de departamento (STC).
rp3_categ	char(5) not null,	Clave de categoría (STC).
rp3_sexo	char(2) not null,	Sexo F) Femenino M) Masculino.
rp3_feact	date,	Fecha de actualización dada por el usuario.
rp3_fecofi	date,	Fecha de oficio.
rp3_refofi	char(10),	Referencia de oficio.
rp3_stent	smallint,	Estatus de cédula de entrega. 1) Entregado.
rp3_sttal	smallint,	Estatus de cédula de tallas. 1) Entregado.
rp3_repent	smallint,	Cantidad de reposiciones de cédula de entrega.
rp3_verent	smallint,	Cantidad de versiones de cédula de entrega.
rp3_reptal	smallint,	Cantidad de reposiciones de cédula de tallas.
rp3_vertal	smallint,	Cantidad de versiones de cédula de tallas.
rp3_flagexc	char(1),	Identificador de excepción E) Excepción.
rp3_motrep	char(1),	Motivo de reposición.
rp3_motcam	char(1),	Motivo de cambio.
rp3_cont	smallint,	Número de prendas correspondientes.
rp3_ident	char(1),	Identificador de vestuario U) Uniforme R) Ropa A) Ambos.
rp3_cverh1	char(4),	Clave asignada en Recursos Humanos (1ra. prenda).
rp3_canti1	char(2),	Cantidad otorgada (1ra. prenda).
rp3_talla1	char(5),	Talla del trabajador (1ra. prenda).
rp3_cveen1	char(4),	Clave asignada en Recursos Humanos entregada (1ra. prenda).
rp3_canen1	char(2),	Cantidad entregada (1ra. prenda)
rp3_cverh2	char(4),	Clave asignada en Recursos Humanos (2da. prenda).
rp3_canti2	char(2),	Cantidad otorgada (2da. prenda).
rp3_talla2	char(5),	Talla del trabajador (2da. prenda).
rp3_cveen2	char(4),	Clave asignada en Recursos Humanos entregada (2da. prenda).
rp3_canen2	char(2),	Cantidad entregada (2da. prenda)
rp3_cverh3	char(4),	Clave asignada en Recursos Humanos (3ra. prenda).
rp3_canti3	char(2),	Cantidad otorgada (3ra. prenda).
rp3_talla3	char(5),	Talla del trabajador (3ra. prenda).
rp3_cveen3	char(4),	Clave asignada en Recursos Humanos entregada (3ra. prenda).
rp3_canen3	char(2),	Cantidad entregada (3ra. prenda)
rp3_cverh4	char(4),	Clave asignada en Recursos Humanos (4a. prenda).
rp3_canti4	char(2),	Cantidad otorgada (4a. prenda).
rp3_talla4	char(5),	Talla del trabajador (4a. prenda).
rp3_cveen4	char(4),	Clave asignada en Recursos Humanos entregada (1ra. prenda).
rp3_canen4	char(2),	Cantidad entregada (4a. prenda)
rp3_cverh5	char(4),	Clave asignada en Recursos Humanos (5a. prenda).
rp3_canti5	char(2),	Cantidad otorgada (5a. prenda).
rp3_talla5	char(5),	Talla del trabajador (5a. prenda).
rp3_cveen5	char(4),	Clave asignada en Recursos Humanos entregada (5a. prenda).
rp3_canen5	char(2),	Cantidad entregada (5a. prenda)
rp3_cverh6	char(4),	Clave asignada en Recursos Humanos (6a. prenda).
rp3_canti6	char(2),	Cantidad otorgada (6a. prenda).

TABLA : ROPA30 MAESTRO DE UNIFOMES Y ROPA DE TRABAJO 2/2

MODULO : UNIFORMES Y ROPA DE TRABAJO EXTENTS: EXTENT SIZE 16 NEXT SIZE 16

rp3_talla6	char(5),	Talla del trabajador (6a. prenda).
rp3_cveen6	char(4),	Clave asignada en Recursos Humanos entregada (6a. prenda).
rp3_canen6	char(2),	Cantidad entregada (6a. prenda)
rp3_cverh7	char(4),	Clave asignada en Recursos Humanos (7a. prenda).
rp3_canti7	char(2),	Cantidad otorgada (7a. prenda).
rp3_talla7	char(5),	Talla del trabajador (7a. prenda).
rp3_cveen7	char(4),	Clave asignada en Recursos Humanos entregada (7a. prenda).
rp3_canen7	char(2),	Cantidad entregada (7a. prenda)
rp3_cverh8	char(4),	Clave asignada en Recursos Humanos (8va. prenda).
rp3_canti8	char(2),	Cantidad otorgada (8va. prenda).
rp3_talla8	char(5),	Talla del trabajador (8va. prenda).
rp3_cveen8	char(4),	Clave asignada en Recursos Humanos entregada (8va. prenda).
rp3_canen8	char(2),	Cantidad entregada (8va. prenda)
rp3_cverh9	char(4),	Clave asignada en Recursos Humanos (9a. prenda).
rp3_canti9	char(2),	Cantidad otorgada (9a. prenda).
rp3_talla9	char(5),	Talla del trabajador (9a. prenda).
rp3_cveen9	char(4),	Clave asignada en Recursos Humanos entregada (9a. prenda).
rp3_canen9	char(2),	Cantidad entregada (9a. prenda)
rp3_cverh10	char(4),	Clave asignada en Recursos Humanos (10a. prenda).
rp3_canti10	char(2),	Cantidad otorgada (10a. prenda).
rp3_talla10	char(5),	Talla del trabajador (10a. prenda).
rp3_cveen10	char(4),	Clave asignada en Recursos Humanos entregada (10a. prenda).
rp3_canen10	char(2),	Cantidad entregada (10a. prenda)
rp3_cverh11	char(4),	Clave asignada en Recursos Humanos (11va. prenda).
rp3_canti11	char(2),	Cantidad otorgada (11va. prenda).
rp3_talla11	char(5),	Talla del trabajador (11va. prenda).
rp3_cveen11	char(4),	Clave asignada en Recursos Humanos entregada (11va. prenda).
rp3_canen11	char(2),	Cantidad entregada (11va. prenda)
rp3_cverh12	char(4),	Clave asignada en Recursos Humanos (12va. prenda).
rp3_canti12	char(2),	Cantidad otorgada (12va. prenda).
rp3_talla12	char(5),	Talla del trabajador (12va. prenda).
rp3_cveen12	char(4),	Clave asignada en Recursos Humanos entregada (12va. prenda).
rp3_canen12	char(2),	Cantidad entregada (12va. prenda)
rp3_cverh13	char(4),	Clave asignada en Recursos Humanos (13va. prenda).
rp3_canti13	char(2),	Cantidad otorgada (13va. prenda).
rp3_talla13	char(5),	Talla del trabajador (13va. prenda).
rp3_cveen13	char(4),	Clave asignada en Recursos Humanos entregada (13va. prenda).
rp3_canen13	char(2),	Cantidad entregada (13va. prenda)
rp3_cverh14	char(4),	Clave asignada en Recursos Humanos (14va. prenda).
rp3_canti14	char(2),	Cantidad otorgada (14va. prenda).
rp3_talla14	char(5),	Talla del trabajador (14va. prenda).
rp3_cveen14	char(4),	Clave asignada en Recursos Humanos entregada (14va. prenda).
rp3_canen14	char(2),	Cantidad entregada (14va. prenda)
rp3_cverh15	char(4),	Clave asignada en Recursos Humanos (15va. prenda).
rp3_canti15	char(2),	Cantidad otorgada (15va. prenda).
rp3_talla15	char(5),	Talla del trabajador (15va. prenda).
rp3_cveen15	char(4),	Clave asignada en Recursos Humanos entregada (15va. prenda).
rp3_canen15	char(2),	Cantidad entregada (15va. prenda)

INDICES

Create unique index "monica".rp3_exej on "monica".ropa30 (rp3_numexp, rp3_ejerc);
Create index "monica".tacs on "monica".ropa30 (rp3_tnomi, rp3_adscr, rp3_categ, rp3_sexo) ;

SISTEMA DE TRANSPORTE COLECTIVO METRO

Sistema de Uniformes y Ropa de Trabajo
Diccionario de Datos:

Base de Datos : "sirh"

TABLA : TABREG RESPALDO DE TALLAS 1/2

		EXTENT SIZE	NEXT SIZE
MODULO : UNIFORMES Y ROPA DE TRABAJO	**EXTENTS:**	16	16

numexp	char(10) not null,	Expediente.
ejerc	char(4) not null,	Ejercicio activo.
feact	date,	Fecha de actualización dada por el usuario.
fecofi	date,	Fecha de oficio.
refofi	char(10),	Referencia de oficio.
stent	smallint,	Estatus de cédula de entrega. 1) Entregado.
sttal	smallint,	Estatus de cédula de tallas. 1) Entregado.
repent	smallint,	Cantidad de reposiciones de cédula de entrega.
verent	smallint,	Cantidad de versiones de cédula de entrega.
reptal	smallint,	Cantidad de reposiciones de cédula de tallas.
vertal	smallint,	Cantidad de versiones de cédula de tallas.
flagexc	char(1),	Identificador de excepción E) Excepción.
motrep	char(1),	Motivo de reposición.
motcam	char(1),	Motivo de cambio.
ident	char(1),	Identificador de vestuario U) Uniforme R) Ropa A) Ambos.
cverh1	char(4),	Clave asignada en Recursos Humanos (1ra. prenda).
canti1	char(2),	Cantidad otorgada (1ra. prenda).
talla1	char(5),	Talla del trabajador (1ra. prenda).
cveen1	char(4),	Clave asignada en Recursos Humanos entregada (1ra. prenda).
canen1	char(2),	Cantidad entregada (1ra. prenda)
cverh2	char(4),	Clave asignada en Recursos Humanos (2da. prenda).
canti2	char(2),	Cantidad otorgada (2da. prenda).
talla2	char(5),	Talla del trabajador (2da. prenda).
cveen2	char(4),	Clave asignada en Recursos Humanos entregada (2da. prenda).
canen2	char(2),	Cantidad entregada (2da. prenda)
cverh3	char(4),	Clave asignada en Recursos Humanos (3ra. prenda).
canti3	char(2),	Cantidad otorgada (3ra. prenda).
talla3	char(5),	Talla del trabajador (3ra. prenda).
cveen3	char(4),	Clave asignada en Recursos Humanos entregada (3ra. prenda).
canen3	char(2),	Cantidad entregada (3ra. prenda)
cverh4	char(4),	Clave asignada en Recursos Humanos (4a. prenda).
canti4	char(2),	Cantidad otorgada (4a. prenda).
talla4	char(5),	Talla del trabajador (4a. prenda).
cveen4	char(4),	Clave asignada en Recursos Humanos entregada (1ra. prenda).
canen4	char(2),	Cantidad entregada (4a. prenda)
cverh5	char(4),	Clave asignada en Recursos Humanos (5a. prenda).
canti5	char(2),	Cantidad otorgada (5a. prenda).
talla5	char(5),	Talla del trabajador (5a. prenda).
cveen5	char(4),	Clave asignada en Recursos Humanos entregada (5a. prenda).
canen5	char(2),	Cantidad entregada (5a. prenda)
cverh6	char(4),	Clave asignada en Recursos Humanos (6a. prenda).
canti6	char(2),	Cantidad otorgada (6a. prenda).
talla6	char(5),	Talla del trabajador (6a. prenda).
cveen6	char(4),	Clave asignada en Recursos Humanos entregada (6a. prenda).
canen6	char(2),	Cantidad entregada (6a. prenda)
cverh7	char(4),	Clave asignada en Recursos Humanos (7a. prenda).
canti7	char(2),	Cantidad otorgada (7a. prenda).
talla7	char(5),	Talla del trabajador (7a. prenda).

SISTEMA DE TRANSPORTE COLECTIVO METRO

Sistema de Uniformes y Ropa de Trabajo
Diccionario de Datos:

Base de Datos : "sirh"

TABLA : TABREG	RESPALDO DE TALLAS		2/2
		EXTENT SIZE	NEXT SIZE
MODULO : UNIFORMES Y ROPA DE TRABAJO	EXTENTS:	16	16

cveen7	char(4),	Clave asignada en Recursos Humanos entregada (7a. prenda).
canen7	char(2),	Cantidad entregada (7a. prenda)
cverh8	char(4),	Clave asignada en Recursos Humanos (8va. prenda).
canti8	char(2),	Cantidad otorgada (8va. prenda).
talla8	char(5),	Talla del trabajador (8va. prenda).
cveen8	char(4),	Clave asignada en Recursos Humanos entregada (8va. prenda).
canen8	char(2),	Cantidad entregada (8va. prenda)
cverh9	char(4),	Clave asignada en Recursos Humanos (9a. prenda).
canti9	char(2),	Cantidad otorgada (9a. prenda).
talla9	char(5),	Talla del trabajador (9a. prenda).
cveen9	char(4),	Clave asignada en Recursos Humanos entregada (9a. prenda).
canen9	char(2),	Cantidad entregada (9a. prenda)
cverh10	char(4),	Clave asignada en Recursos Humanos (10a. prenda).
canti10	char(2),	Cantidad otorgada (10a. prenda).
talla10	char(5),	Talla del trabajador (10a. prenda).
cveen10	char(4),	Clave asignada en Recursos Humanos entregada (10a. prenda).
canen10	char(2),	Cantidad entregada (10a. prenda)
cverh11	char(4),	Clave asignada en Recursos Humanos (11va. prenda).
canti11	char(2),	Cantidad otorgada (11va. prenda).
talla11	char(5),	Talla del trabajador (11va. prenda).
cveen11	char(4),	Clave asignada en Recursos Humanos entregada (11va. prenda).
canen11	char(2),	Cantidad entregada (11va. prenda)
cverh12	char(4),	Clave asignada en Recursos Humanos (12va. prenda).
canti12	char(2),	Cantidad otorgada (12va. prenda).
talla12	char(5),	Talla del trabajador (12va. prenda).
cveen12	char(4),	Clave asignada en Recursos Humanos entregada (12va. prenda).
canen12	char(2),	Cantidad entregada (12va. prenda)
cverh13	char(4),	Clave asignada en Recursos Humanos (13va. prenda).
canti13	char(2),	Cantidad otorgada (13va. prenda).
talla13	char(5),	Talla del trabajador (13va. prenda).
cveen13	char(4),	Clave asignada en Recursos Humanos entregada (13va. prenda).
canen13	char(2),	Cantidad entregada (13va. prenda)
cverh14	char(4),	Clave asignada en Recursos Humanos (14va. prenda).
canti14	char(2),	Cantidad otorgada (14va. prenda).
talla14	char(5),	Talla del trabajador (14va. prenda).
cveen14	char(4),	Clave asignada en Recursos Humanos entregada (14va. prenda).
canen14	char(2),	Cantidad entregada (14va. prenda)
cverh15	char(4),	Clave asignada en Recursos Humanos (15va. prenda).
canti15	char(2),	Cantidad otorgada (15va. prenda).
talla15	char(5),	Talla del trabajador (15va. prenda).
cveen15	char(4),	Clave asignada en Recursos Humanos entregada (15va. prenda).
canen15	char(2),	Cantidad entregada (15va. prenda)

I N D I C E S

Create unique index "monica".exej on "monica".tabreg (numexp,ejerc)

SISTEMA DE TRANSPORTE COLECTIVO METRO

Sistema de Uniformes y Ropa de Trabajo
Diccionario de Datos:

Base de Datos : "sirh"

TABLA : TABROP	REPOSICIONES	EXTENT SIZE	NEXT SIZE
MODULO : UNIFORMES Y ROPA DE TRABAJO	EXTENTS:	16	16

numexp	char(10) not null,	Expediente.
motrep	char(30)	Motivo de reposición.

I N D I C E S

Create unique index "monica".ix1472_1 on "monica".tabrop (numexp)

SISTEMA DE TRANSPORTE COLECTIVO METRO

Sistema de Uniformes y Ropa de Trabajo
Diccionario de Datos:

Base de Datos : "sirh"

TABLA : TALABI OPCION ABIERTA DE CEDULAS DE TALLAS

	EXTENT SIZE	NEXT SIZE
MODULO : UNIFORMES Y ROPA DE TRABAJO	**EXTENTS:** 16	16

ejerc	char(4),	Ejercicio activo.
numexp	char(10),	Expediente.
nombre	char(50),	Nombre del empleado.
sexo	char(2),	Sexo F) Femenino M) Masculino.
tipnom	char(1),	Tipo de nómina 1) Base 2) Confianza 3) Eventual 4) Base Complementaria
adscri	char(8),	Clave de departamento (STC).
catego	char(5),	Clave de categoría (STC).
idvest	char(1),	Identificador de vestuario U) Uniforme R) Ropa A) Ambos
idexce	char(1),	Identificador de excepción E) Excepción
cverh1	char(4),	Clave asignada en Recursos Humanos (1ra. prenda).
canti1	char(2),	Cantidad otorgada (1ra. prenda).
cverh2	char(4),	Clave asignada en Recursos Humanos (2da. prenda).
canti2	char(2),	Cantidad otorgada (2da. prenda).
cverh3	char(4),	Clave asignada en Recursos Humanos (3ra. prenda).
canti3	char(2),	Cantidad otorgada (3ra. prenda).
cverh4	char(4),	Clave asignada en Recursos Humanos (4a. prenda).
canti4	char(2),	Cantidad otorgada (4a. prenda).
cverh5	char(4),	Clave asignada en Recursos Humanos (5a. prenda).
canti5	char(2),	Cantidad otorgada (5a. prenda).
cverh6	char(4),	Clave asignada en Recursos Humanos (6a. prenda).
canti6	char(2),	Cantidad otorgada (6a. prenda).
cverh7	char(4),	Clave asignada en Recursos Humanos (7a. prenda).
canti7	char(2),	Cantidad otorgada (7a. prenda).
cverh8	char(4),	Clave asignada en Recursos Humanos (8va. prenda).
canti8	char(2),	Cantidad otorgada (8va. prenda).
cverh9	char(4),	Clave asignada en Recursos Humanos (9a. prenda).
canti9	char(2),	Cantidad otorgada (9a. prenda).
cverh10	char(4),	Clave asignada en Recursos Humanos (10a. prenda).
canti10	char(2),	Cantidad otorgada (10a. prenda).
cverh11	char(4),	Clave asignada en Recursos Humanos (11va. prenda).
canti11	char(2),	Cantidad otorgada (11va. prenda).
cverh12	char(4),	Clave asignada en Recursos Humanos (12va. prenda).
canti12	char(2),	Cantidad otorgada (12va. prenda).
cverh13	char(4),	Clave asignada en Recursos Humanos (13va. prenda).
canti13	char(2),	Cantidad otorgada (13va. prenda).
cverh14	char(4),	Clave asignada en Recursos Humanos (14va. prenda).
canti14	char(2),	Cantidad otorgada (14va. prenda).
cverh15	char(4),	Clave asignada en Recursos Humanos (15va. prenda).
canti15	char(2)	Cantidad otorgada (15va. prenda).

I N D I C E S

Sin índices.

APENDICE C

```
-- ****************************************
-- * SISTEMA   : Uniformes y ropa de trabajo *
-- * PROGRAMA  : ropemenu.4gl                *
-- * OBJETIVO  : Menu principal              *
-- * AUTOR     : Monica Ramirez Hernandez    *
-- * FECHA     : 19 de julio de 1996         *
-- ****************************************

database siru
GLOBALS
    "ropglb.4gl"

MAIN
   DEFER INTERRUPT
       call set_option()
       call main_inic()
   {    run "banner 'Uniformes'"
       run "banner 'y Ropa de'"
       run "banner ' Trabajo '"
       sleep 4 }
       call main_menu()
       clear screen
END MAIN

FUNCTION set_option()
   OPTIONS
       ERROR    LINE 24,
       HELP     FILE "menuayu.ex",
       HELP     KEY CONTROL-W,
       DELETE KEY CONTROL-B,
       ACCEPT KEY ESC
         INPUT  WRAP
END FUNCTION

FUNCTION main_menu()
   MENU "MENU PRINCIPAL"
      COMMAND "Actualizaciones Catalogos"
              "Menu     Mantenimiento a Catalogos"
              call catalo()

      COMMAND "Generacion/Captura"
              "Menu  Genera Mto. ropa y Cedulas / Captura Tallas y Prendas"
              call proces()

      COMMAND "Presupuesto"
              "Menu      Informacion Estadistica"
              call presup()

      COMMAND "Reportes"
              "Menu  Reportes Varios"
              call repo()

      COMMAND "Salir"
              "Salir del Sistema"
              clear screen
      EXIT MENU
   END MENU
END FUNCTION

FUNCTION repo()
   MENU "REPORTES DE ROPA"
      COMMAND "rop02r"
              "(rop02r)    Catalogo de Prendas"
              call rop02r()

      COMMAND "rop04r"
              "(rop04r)    Asignacion de Ropa por Categoria"
              call rop04r()

      COMMAND "rop05r"
              "(rop05r)    Cantidad de Personal Calificado para Ropa"
              call rop05r()

      COMMAND "rop06r"
              "(rop06r)    Catalogo con Homologacion de Dotaciones"
              call rop06r()

      COMMAND "rop07r"
              "(rop07r)    Personal Incluido en Excepciones"
              call rop07r()

      COMMAND "rop07a"
              "(rop07a)    Excepciones Con Nombre y Cve. cobro"
              call rop07a()

      COMMAND "rop10r"
              "(rop10r)    Categorias No Calificadas para Ropa"
              call rop10r()

      COMMAND "rop11r"
              "(rop11r)    Empleados no Calificados para Ropa"
              call rop11r()

      COMMAND "rop14r"
              "(rop14r)    Validacion de Captura de Tallas"
              call rop14r()

      COMMAND "rop20r"
              "(rop20r)    Validacion De Ropa Ya Entregada"
              call rop20r()

      COMMAND "rop39r"
              "(rop39r)    Dotaciones TOTALMENTE Entregadas"
              call rop39r()

      COMMAND "rop40r"
              "(rop40r)    Dotaciones PARCIALMENTE Entregadas"
              call rop40r()

      COMMAND "rop41r"
              "(rop41r)    Dotaciones PENDIENTES de Entregar"
              call rop41r()

      COMMAND "rop25r"
```

```
            "(rop25r)    Citas Totales por Tipo de Prenda"
             call rop25r()

      COMMAND "rop26r"
            "(rop26r)    Citas Generales por Tipo de Prenda "
             call rop26r()

      COMMAND "rop31r"
            "(rop31r)    Empleados  con Tallas en Blanco"
             call rop31r()

      COMMAND "rop36r"
            "(rop16r)    Empleados  con una o varias Tallas en Blanco"
             call rop36r()

      COMMAND "Salir" "Salir del Sistema"
            clear screen
      EXIT MENU
   END MENU
END FUNCTION

FUNCTION cat1o()
   MENU "MANTO. A CATALOGOS"
      COMMAND "Catalogo de Prendas"
            "(Menu)    Mantenimiento (Altas,Bajas,Cambios,Consultas)"
            call cat10()

      COMMAND "Catalogo de Categorias"
            "(Menu)    Mantenimiento (Altas,Bajas,Cambios,Consultas)"
            call cat20()

      COMMAND "Archivo Maestro de Ropa"
            "(Menu)    Mantenimiento (Altas,Bajas,Cambios,Consultas)"
            call cat30()

      COMMAND "Consulta a Empleado"
            "Datos Generales del Empleado"
            call rop35c()

      COMMAND "Salir" "Salir del Sistema"
            clear screen
      EXIT MENU
   END MENU
END FUNCTION

FUNCTION cat10()
   MENU "MANTO. A CATALOGO DE PRENDAS"
      COMMAND "Actualizacion"
            "(rop01m)    Altas, Bajas y Cambios"
            call rop01m()

      COMMAND "Consultas"
            "(rop01c)    Consulta  por  Criterio"
            call rop01c()

      COMMAND "Salir" "Salir del Sistema"
            clear screen
      EXIT MENU
   END MENU
END FUNCTION

FUNCTION cat20()
   MENU "MANTO. A CATALOGO DE CATEGORIAS"
      COMMAND "Actualizacion"
            "(rop03m)    Altas, Bajas y Cambios"
            call rop03m()

      COMMAND "Consultas"
            "(rop03c)    Consultas por Criterio"
            call rop03c()

      COMMAND "Salir" "Salir del Sistema"
            clear screen
      EXIT MENU
   END MENU
END FUNCTION

FUNCTION cat30()
   MENU "MANTO. A ARCHIVO MAESTRO DE ROPA"
      COMMAND "Actualizacion"
            "(rop09m)    Altas, Bajas y Cambios"
            call rop09m()

      COMMAND "Consultas"
            "(rop09c)    Consultas por Criterio"
            call rop09c()

      COMMAND "Salir" "Salir del Sistema"
            clear screen
      EXIT MENU
   END MENU
END FUNCTION

FUNCTION proces()
   MENU "PROCESOS VARIOS"
      COMMAND "Archivo Maestro de Ropa"
            "(rop08m)    Generacion catorcenal"
            call rop08m()

      COMMAND "Cedulas de Recopilacion Tallas"
            "(Menu)    Emision Individual, Horizontal, Especial, Rangos"
            call proc1()

      COMMAND "Tallas"
            "(rop15m)    Captura de Tallas"
            call rop15m()

      COMMAND "Cedulas de Entrega"
            "(Menu)    Emision Individual, Horizontal, Especial, Rangos"
            call proc4()

      COMMAND "Prendas Entregadas"
            "(rop20m)    Captura de Prendas Entregadas"
            call rop20m()
```

```
      COMMAND "Salir" "Salir del Sistema"
              clear screen
      EXIT MENU
   END MENU
END FUNCTION

FUNCTION proenc()
   MENU "CEDULAS DE ENTREGA"
      COMMAND "Generacion"
              "(rop197)   Emision Cedulas Individual"
              call rop197()

      COMMAND "Listado"
              "(rop180)  Emision en Forma Horizontal"
              call rop180()

      COMMAND "Opcion Abierta"
              "(menu)   Opcion para cedulas de Emision Especial"
              call opabl()

      COMMAND "Reposicion"
              "(menu)   Opciones por Reposicion"
              call repcent()

      COMMAND "Salir" "Salir del Sistema"
             clear screen
             EXIT MENU
   END MENU
END FUNCTION

FUNCTION opabl()
   MENU "CEDULAS DE ENTREGA"
      COMMAND "Captura"
              "(rop314)   Captura Para Emision Especial de Cedulas"
              call rop314()

      COMMAND "Listado"
              "(rop24r)  Emision Especial de Cedulas"
              call rop24r()

      COMMAND "Salir" "Salir del Sistema"
             clear screen
             EXIT MENU
   END MENU
END FUNCTION

FUNCTION protal()
   MENU "CEDULAS DE TALLAS"
      COMMAND "Generacion"
              "(rop12r)   Emision Cedulas"
              call rop12r()

      COMMAND "Listado"
              "(rop13r)  Emision en Forma Horizontal"
              call rop13r()

      COMMAND "Opcion Abierta"
              "(Menu)   Opcion para cedulas de tallas de Emision Especial"
              call opabll()

      COMMAND "Reposicion"
              "(Menu)   Opciones por Reposicion"
              call menrep()

      COMMAND "Salir" "Salir del Sistema"
             clear screen
             EXIT MENU
   END MENU
END FUNCTION

FUNCTION opabll()
   MENU "CEDULAS DE TALLAS"
      COMMAND "Captura"
              "(rop323m)   Captura Para Emision Especial de Cedulas de Tallas"
              call rop323m()

      COMMAND "Listado"
              "(rop23r)  Emision Especial de Cedules de Tallas"
              call rop23r()

      COMMAND "Salir" "Salir del Sistema"
             clear screen
             EXIT MENU
   END MENU
END FUNCTION

FUNCTION repcent()
   MENU "REPOSICION DE CEDULAS DE ENTREGA"
      COMMAND "Por Rangos"
              "(rop19r)    Emision (adscr,categoria,tipo nomina,exp)"
              call rop19r()

      COMMAND "Por Expedientes"
              "(rop19c)    Emision (Expedientes)"
              call rop19c()

      COMMAND "Salir" "Salir del Sistema"
              clear screen
      EXIT MENU
   END MENU
END FUNCTION

FUNCTION menrep()
   MENU "REPOSICION DE CEDULAS DE TALLAS"
      COMMAND "Por Rangos"
              "(rop14r)    Emision (adscr,categoria,tipo nomina,exp)"
              call rop14r()

      COMMAND "Por Expedientes"
              "(rop14c)    Emision (Expedientes)"
              call rop14c()
```

```
      COMMAND "Salir" "Salir del Sistema"
              clear screen
      EXIT MENU
   END MENU
END FUNCTION

FUNCTION presup()
   MENU "PRESUPUESTO"
      COMMAND "Detalle"
              "(rop21r)    Genera Informacion detallada"
              call rop21r()

      COMMAND "Cifras Totales"
              "(rop20r)    Conteo por tipo de prendas segun plantilla"
              call rop20r()

      COMMAND "Cifras Generales"
              "(rop10r)    Resumen de conteo de prendas segun plantilla"
              call rop10r()

      COMMAND "Cifras Adscripcion G."
              "(rop29r)    Genera Cifras Generales por Adscripcion"
              call rop29r()

      COMMAND "Cifras Adscripcion T."
              "(rop29a)    Genera Cifras Totales por Adscripcion"
              call rop29a()

      COMMAND "Cifras Totales por Tallas"
              "(rop27r)    Genera Cifras Totales de Prendas por Tallas"
              call rop27r()

      COMMAND "Cifras Generales por Tallas"
              "(rop22r)    Resumen de Cifras Generales de Prendas por Tallas"
              call rop22r()

      COMMAND "Cifras de Prendas sin Tallas"
              "(rop1er)    Resumen de Cifras de Prendas con Tallas en Blanco"
              call rop1er()

      COMMAND "Prendas Entregadas"
              call menentreg()

      COMMAND "Excepciones"
              call menexc()


      COMMAND "Salir" "Salir del Sistema"
              clear screen
      EXIT MENU
   END MENU
END FUNCTION

FUNCTION menentreg()
   MENU "ENTREGADAS"
      COMMAND "TOTAL"
              "(rop42r)    Resumen de Prendas Entregadas Totalmente"
              call rop42r()
      COMMAND "PARCIAL"
              "(rop43r)    Resumen de Prendas Entregadas Parcialmente"
              call rop43r()
      COMMAND "PENDIENTES"
              "(rop44r)    Resumen de Prendas Pendientes de Entregar"
              call rop44r()
      COMMAND "Salir" "Salir del Sistema"
              clear screen
      EXIT MENU
   END MENU
END FUNCTION

FUNCTION menexc()
   MENU "EXCEPCIONES"
      COMMAND "rop17r"
              "(rop17r)    Resumen de Cifras Generales de Tallas"
              call rop17r()
      COMMAND "rop18r"
              "(rop18r)    Cifras Generales por Tipo de Prenda"
              call rop18r()
      COMMAND "Salir" "Salir del Sistema"
             clear screen
             EXIT MENU
   END MENU
END FUNCTION

FUNCTION d_*my()
   ERROR "Esta opcion aun no ha sido desarrollada" ATTRIBUTE (REVERSE, BLINK)
END FUNCTION
```

```
-- ****************************************
-- * SISTEMA  : Uniformes y ropa de trabajo *
-- * PROGRAMA : capexp.4gl                  *
-- * OBJETIVO : Altas en tabla tabrop       *
-- * AUTOR    : Monica Ramirez Hernandez    *
-- * FECHA    : 24 de julio de 1996         *
-- ****************************************

GLOBALS
   "ropglb.4gl"
DEFINE dr_numexp CHAR(10),motrep char(1)

FUNCTION capexp()

   ERROR "Favor de Esperar, se esta limpiando la tabla..."
   SLEEP 2
   DELETE FROM tabrop

   CLEAR SCREEN

   OPEN FORM fropa FROM "fcapexp"
   DISPLAY FORM fropa
   WHILE true

      INPUT BY NAME dr_numexp,motrep

         AFTER FIELD dr_numexp
            IF dr_numexp is null or dr_numexp = " " THEN
               clear form
                  error "Expediente invalido. Verifique  . "
                  next field dr_numexp
            ELSE
               let dr_numexp = dr_numexp using "&&&&&&&&&&"
               DISPLAY BY NAME dr_numexp
            END IF

      END input

      IF int_flag = TRUE THEN
         let int_flag = FALSE
         CLEAR FORM
         ERROR "PROCESO TERMINADO .. "
         sleep 2
         clear screen
         exit WHILE
         return
      ELSE
         let dr_numexp = dr_numexp using "&&&&&&&&&&"
         INSERT INTO tabrop (numexp,motrep)
         VALUES (dr_numexp,motrep)

      END IF

      IF SQLCA.SQLCODE < 0 THEN
         ERROR "PROBLEMAS EN LA ACTUALIZACION..."
      ELSE
         ERROR "EXPEDIENTE AGREGADO CON EXITO . "
         clear form
      END IF

   END WHILE
END FUNCTION
```

```
-- *****************************************
-- * SISTEMA   : Uniformes y ropa de trabajo *
-- * PROGRAMA  : rop01c 4gl                  *
-- * OBJETIVO  : Consultas por criterios     *
-- *             de ropa10                   *
-- * AUTOR     : Monica Ramirez Hernandez    *
-- * FECHA     : 23 de agosto de 1996        *
-- *****************************************

-----------------------------------------------------------------
--              DECLARACION DE VARIABLES GLOBALES             --
-----------------------------------------------------------------

GLOBALS
        "ropglo.4gl"

define gl_ropa10 record like ropa10.*

-----------------------------------------------------------------
--              DECLARACION DE VARIABLES LOCALES              --
-----------------------------------------------------------------

 DEFINE
                ropa10_query    CHAR(300),
                ropa10_count    CHAR(300),
                ropa10_cnt      SMALLINT,
                where_clause    CHAR(250),
                fetch_flag      SMALLINT,
                win_flag        SMALLINT

-----------------------------------------------------------------
-   prueba. RUTINA QUE REALIZA EL SUB-MENU DE BAJAS ropa10--
-----------------------------------------------------------------

FUNCTION rop01c()

  OPEN WINDOW w_ropa106 AT 3,2 WITH 2 ROWS, 74 COLUMNS ATTRIBUTE(BORDER)
        MENU "MENU CONSULTAS"
                BEFORE MENU
                HIDE OPTION ALL
                SHOW OPTION "1. Consultas","5. Salir"
                COMMAND KEY ("1",C) "1. Consultas"
                        "Consulta de ropa por criterio"
                   IF pruebaf() = TRUE THEN
                                HIDE OPTION ALL
                                SHOW OPTION "2. siGuiente","3. Previo","5. Salir"
                        ELSE
                                HIDE OPTION ALL
                                SHOW OPTION "1. Consultas","5. Salir"
                        END  IF

                COMMAND KEY ("2",G) "2. siGuiente"
                        "Muestra el Registro siguiente"
                        CALL apuntar(1)

                COMMAND KEY ("3",P) "3. Previo"
                        "Muestra el Registro Previo"
                        CALL apuntar(-1)

                COMMAND KEY ("5",S) "5. Salir"
                        "Regresar a Opciones de Menu de Consultas"
                        CALL clean_ropa10()
                        IF win_flag = 1 THEN
                                CLOSE WINDOW w_ropa105
                                LET win_flag = 0
                        END IF
                        EXIT MENU
        END MENU
        CLOSE WINDOW w_ropa106
END FUNCTION

-----------------------------------------------------------------
-- pruebaf.RUTINA QUE MUESTRA EL REGISTRO PARA PLAZAS--
-----------------------------------------------------------------

 FUNCTION pruebaf()

        OPEN WINDOW w_ropa105 AT 4,1 WITH form "frop01c"

        LET win_flag = 1
        LET int_flag = FALSE
   CONSTRUCT BY NAME where_clause ON ropa10.*
                BEFORE CONSTRUCT
                DISPLAY "Presione <Esc> para presentar informacion del catalogo ...","" AT 1,1
                DISPLAY "Presione <PAUSA> para cancelar o salir....", "" AT 2,1
        END CONSTRUCT

        IF int_flag = TRUE THEN
                LET int_flag = FALSE
                CLEAR FORM
                ERROR "Proceso de consulta CANCELADO...."
                SLEEP 1
                CLOSE WINDOW w_ropa105
                LET win_flag = 0
                RETURN FALSE
         ELSE

                LET ropa10_query = "SELECT * FROM ropa10 WHERE ",where_clause CLIPPED
                LET ropa10_count = "SELECT COUNT(*) FROM ropa10 WHERE ", where_clause CLIPPED

                PREPARE ropa10_stmt  FROM ropa10_query
                PREPARE ropa10_stmt2 FROM ropa10_count

                DECLARE ropa10_con SCROLL CURSOR FOR ropa10_stmt

                -  OPEN ropa10_stmt
                OPEN ropa10_con

                FETCH FIRST ropa10_con INTO gl_ropa10.*

                IF SQLCA.SQLCODE = NOTFOUND THEN
                        ERROR "No existen registros que cumplan la condic.on...."
                        LET win_flag = 0
                        CLOSE WINDOW w_ropa105
```

```
                        CLOSE ropa10_con
                        FREE  ropa10_con
                        RETURN FALSE
                ELSE
                        DECLARE ropa102_prt CURSOR FOR ropa10_stmt2
                        OPEN    ropa102_prt
                        FETCH   ropa102_prt INTO ropa10_cnt
                        CLOSE   ropa102_prt
                        FREE    ropa102_prt
                        CALL display_ropa10()
                        #MESSAGE "Existen ", ropa10_cnt USING "<<<<", " registros..."
                        RETURN TRUE
                END IF
        END IF
        CLOSE WINDOW w_ropa105

END FUNCTION

--------------------------------------------------------------------
-- apuntar.RUTINA 2: VERIFICA PRIMER Y ULTIMO REGIS  TABLA TEMP.  --
--------------------------------------------------------------------

FUNCTION apuntar(fetch_flag)

                DEFINE fetch_flag SMALLINT
                FETCH RELATIVE fetch_flag ropa10_con INTO gl_ropa10.*

                IF SQLCA.SQLCODE = NOTFOUND THEN
                        IF fetch_flag = 1 THEN
                                ERROR "Esta ubicado al final de la tabla..."
                        ELSE
                                ERROR "Esta ubicado al principio de la tabla..."
                        END IF
                ELSE
                        CURRENT WINDOW is w_ropa105
                        CALL display_ropa10()
                END IF

END FUNCTION

--------------------------------------------------------------------
--   display_2u.  RUTINA PARA DESPLEGAR LOS VALORES DEL CATALOGO  --
--------------------------------------------------------------------

FUNCTION display_ropa10()

                DISPLAY BY NAME gl_ropa10.*

END FUNCTION

--------------------------------------------------------------------
--   clean_2010  RUTINA    PARA LIMPIAR AREA DE CONSULTAS         --
--------------------------------------------------------------------

 FUNCTION clean_ropa10()

                WHENEVER ERROR CONTINUE
                CLOSE ropa10_con
                FREE  ropa10_con
                WHENEVER ERROR STOP

 END FUNCTION
```

```
-- ****************************************
-- * SISTEMA  : Uniformes y ropa de trabajo *
-- * PROGRAMA : rop01w.4gl                  *
-- * OBJETIVO : Actualizacion a catalogo de *
-- *            claves ropa10               *
-- * AUTOR    : Monica Ramirez Hernandez    *
-- * FECHA    : 27 de agosto de 1996        *
-- ****************************************

GLOBALS
   "rop01b.4gl"

define cveh char(04)
define cves char(08)
define resp char(01)
define respl char(01)

FUNCTION rop01w()

   OPEN FORM frop FROM "frop01w"
   DISPLAY FORM frop
   display "<CONTROL-B> BAJA" at 18,30
while true
     let resp = " "
     let respl = " "

     INPUT BY NAME gr_ropa1.*

      AFTER FIELD rpl_cverh
                  if gr_ropa1.rpl_cverh is null then
                  clear form
                        error "Cve. recuerde Numeros en blanco no es valida..."
                        next field rpl_cverh
               else
                  let gr_ropa1.rpl_cverh = upshift(gr_ropa1.rpl_cverh using "####")
                  select * into gr_ropa1 * from ropa10
                  where ropa10.rpl_cverh = gr_ropa1.rpl_cverh
                  if sqlca.sqlcode = 100 then
                     open window ventana5 at 12,10 with 1 rows,50 columns
                        ATTRIBUTE(BORDER)
                     prompt "Clave no existente, Desea dar alta (S/N)? "
                           for resp
                     close window ventana5
                     if upshift(resp) = "N" then
                       initialize gr_ropa1.* to null
                       next field rpl_cverh
                     end if
                  else
                     display by name gr_ropa1.*
                  end if
                      end if

      AFTER FIELD rpl_cveal
        if gr_ropa1.rpl_cveal is null then
           error "Cve. almacen en blanco no es valida..."
           next field rpl_cveal
        else
           let gr_ropa1.rpl_cveal = gr_ropa1.rpl_cveal using "&&&&&&&&"
        end if

      AFTER FIELD rpl_desc
        if gr_ropa1.rpl_desc is null  or gr_ropa1.rpl_desc = " " then
           error "Descripcion en blanco no es valida..."
           next field rpl_desc
        end if

     on key (CONTROL-B)
       open window ventana1 at 12,10 with 1 rows,50 columns
       ATTRIBUTE(BORDER)
       prompt "Confirme la baja del registro ... (S/N) "
       for respl
       close window ventana1
       if upshift(respl) = "S" then
          DELETE FROM ropa10
          where ropa10.rpl_cverh = gr_ropa1.rpl_cverh
          ERROR "Se realizo baja en catalogo de ropa    "
       end if
          clear form
          initialize gr_ropa1.* to null
          next field rpl_cverh
   end input

      IF int_flag = TRUE THEN
         let int_flag = FALSE
         CLEAR FORM
         ERROR "ACTUALIZACION CANCELADA ..."
         sleep 2
         clear screen
                        exit while
         return
      ELSE
         if upshift(resp) = "S" then
           INSERT INTO ropa10 values(gr_ropa1.*)
         else
           UPDATE ropa10 set rpl_cverh = gr_ropa1.rpl_cverh,
                        rpl_cveal = gr_ropa1.rpl_cveal,
                        rpl_desc  = gr_ropa1.rpl_desc
           WHERE ropa10.rpl_cverh = gr_ropa1.rpl_cverh
         end if

      END IF


      if SQLCA.SQLCODE = 0 then
         ERROR "PROBLEMAS EN LA ACTUALIZACION  ."
      else
         ERROR "ACTUALIZACION EFECTUADA .. "
         clear form
      end if

end while
END FUNCTION
```

```
-- *****************************************
-- * SISTEMA  : Uniformes y ropa de trabajo  *
-- * PROGRAMA : rop02r.4gl                   *
-- * OBJETIVO : Reporte del catalogo de prendas *
-- *            tabla ropa10                 *
-- * AUTOR    : Monica Ramirez Hernandez     *
-- * FECHA    : 17 de julio de 1996          *
-- *****************************************

database sirh
globals
define lr_02r record
       rpl_cverh  like ropa10.rpl_cverh,
       rpl_cveal  like ropa10.rpl_cveal,
       rpl_desc   like ropa10.rpl_desc
end record
end globals

FUNCTION rop02r
define vm_numrma    smallint
define vm_numcmacha char(2)
define vm_nomarch   varchar(20)
define vm_nombre    varchar(100)
define vm_comando   varchar(50)
            MENU "Enviar Reporte a "
                    COMMAND "Impresora" "Envia Reporte a la Impresora"
                            call crea_nomarch("rop02r") returning vm_nomarch
                               call imp_imp(10,10,vm_nomarch) returning vm_comando
                               START REPORT rep_02r TO vm_nomarch
                               CALL rop02ra()
                               call manda_impre(vm_comando,vm_nomarch)
                               call borra_arch(vm_nomarch)
                    COMMAND "Archivo" "Envia Reporte a un Archivo"
                               call pide_nombre(10,10) returning vm_nombre
                               if vm_nombre is null then
                                       next option "Salida"
                               else
                           ERROR "Favor de esperar...Gracias"
                                    START REPORT rep_02r TO vm_nombre
                                    CALL rop02ra()
                                    next option "Reimpresion"
                               end if
                    COMMAND "Pantalla" "Envia Reporte a la Pantalla"
                     ERROR "Favor de esperar...Gracias"
                               call crea_nomarch("rop02r") returning vm_nomarch
                               START REPORT rep_02r TO vm_nomarch
                               CALL rop02ra()
                               call mare_archivo(vm_nomarch)
                    COMMAND "Reimpresion" "Envia a impresora el reporte generado en la opcion <Archivo>"
                               call pide_nombre(10,10) returning vm_nombre
                               if vm_nombre is null then
                                       next option "Salida"
                               else
                                  call imp_imp(10,10,vm_nombre) returning vm_comando
                                  call manda_impre(vm_comando,vm_nombre)
                                  call borra_arch(vm_nombre)
                               end if
                    COMMAND "Salida" "Salir al Menu Principal"
                               EXIT MENU
       END MENU
END FUNCTION

function rop02ra()
    declare cur_rop02r cursor for
    select  rpl_cverh,rpl_cveal,rpl_desc
    from    ropa10
    --start report rep_02r to "/users/produ/reportes/rop02r.rep"
      --    error "generando reporte /users/produ/reportes/rop02r.rep"
      --    sleep 5
    foreach cur_rop02r into lr_02r.*
            output to report rep_02r(lr_02r.*)
    end foreach
       finish report rep_02r
      { error "FAVOR DE CONFIGURAR LA IMPRESORA A LETRA GRANDE"
       sleep 1 }
       clear screen
end function

report rep_02r(r125)
define r125 record
       rpl_cverh  like ropa10.rpl_cverh,
       rpl_cveal like ropa10.rpl_cveal,
       rpl_desc like ropa10.rpl_desc
end record
{
output
   left margin 0
}
order by r125.rpl_cverh
format
page header
     print column  2, "rop02r.rep",
           column 17, "SISTEMA DE TRANSPORTE COLECTIVO (METRO)",
           column 60, "FECHA : ",today
     print column  2, "DIR. DE ADMON.",
           column 25, "GERENCIA DE RECURSOS HUMANOS",
           column 60, "PAG.  : ", pageno using "###"
     print column  1, "UNIFORMES Y ROPA DE TRABAJO",
           column 60, "HORA  :  ",time
     skip 1 line
     print column 25, "CATALOGO DE PRENDAS DE ROPA"
     print "-------------------------------------------------------------------------------"
     print column  6, "CVE.REC.HUM.",
           column 28, "CVE.ALMACEN",
           column 57, "DESCRIPCION"
     print "-------------------------------------------------------------------------------"
on every row
     print column 10, r125.rpl_cverh,
           column 30, r125.rpl_cveal,
           column 45, r125.rpl_desc
on last row
     skip 1 line
       print column 10, "TOTAL REGISTROS IMPRESOS : ", count(*) using "#,###"
end report
```

```
-- ********************************************
-- * SISTEMA  : Uniformes y ropa de trabajo   *
-- * PROGRAMA : rop03c.4gl                    *
-- * OBJETIVO : Consultas al catalogo de categorias *
-- * AUTOR    : Monica Ramirez Hernandez      *
-- * FECHA    : 12 de agosto de 1996          *
-- ********************************************

GLOBALS
   "rop01b.4gl"

DEFINE gl_ropa20 record like ropa20.*

DEFINE
   ropa20_query    CHAR(300),
   ropa20_count    CHAR(300),
   rop020_cnt      SMALLINT,
   where_clause    CHAR(550),
   fetch_flag      SMALLINT,
   win_flag        SMALLINT

FUNCTION rop03c()

   OPEN WINDOW w_ropa206 AT 2,2 WITH 2 ROWS, 74 COLUMNS ATTRIBUTE(BORDER)
      MENU "MENU CONSULTAS"
         BEFORE MENU
                HIDE OPTION ALL
                SHOW OPTION "1. Consultas","5. Salir"
                   COMMAND KEY ("1",C) "1. Consultas"
                           "Consulta de ropa por criterio"
                         IF rop03cf() = TRUE THEN
                            HIDE OPTION ALL
                            SHOW OPTION "2. Siguiente" "3. Previo" "5. Salir"
                         ELSE
                            HIDE OPTION ALL
                            SHOW OPTION "1. Consultas","5. Salir"
                         END IF

                   COMMAND KEY ("2",G) "2. Siguiente"
                                       "Muestra el Registro Siguiente"
                                       CALL apun20(1)

                   COMMAND KEY ("3",P) "3. Previo"
                                       "Muestra el Registro Previo"
                                       CALL apun20(-1)

                   COMMAND KEY ("5",S) "5. Salir"
                                       "Regresar a Opciones de Menu de Consultas"
                                       CALL clean_ropa20()
                                       IF win_flag = 1 THEN
                                          CLOSE WINDOW w_ropa205
                                          LET win_flag = 0
                                       END IF
         EXIT MENU
      END MENU
   CLOSE WINDOW w_ropa206
END FUNCTION

FUNCTION rop03cf()

   OPEN WINDOW w_ropa205 AT 5,1 WITH form "frop03c"

      LET win_flag = 1
      LET int_flag = FALSE

      MESSAGE  "                    "

      CONSTRUCT BY NAME where_clause ON ropa20.*
            BEFORE CONSTRUCT
               DISPLAY "Introducir criterio de consulta... Presione <ENTER>.","" AT 1,1
               DISPLAY "Presione <PAUSA> para cancelar o salir....", "" AT 2,1
      END CONSTRUCT
      IF int_flag = TRUE THEN
            LET int_flag = FALSE
            CLEAR FORM
            ERROR "Proceso de consulta CANCELADO...."
            SLEEP 1
            CLOSE WINDOW w_ropa205
            LET win_flag = 0
            RETURN FALSE
      ELSE
            LET ropa20_query = "SELECT * FROM ropa20 WHERE ", where_clause CLIPPED
         LET ropa20_count = "SELECT COUNT(*) FROM ropa20 WHERE ",
            where_clause CLIPPED
            PREPARE ropa20_stmt  FROM ropa20_query
            PREPARE ropa20_stmt2 FROM ropa20_count
            DECLARE ropa20_con SCROLL CURSOR FOR ropa20_stmt
            OPEN ropa20_con
            FETCH FIRST ropa20_con INTO gl_ropa20.*
            IF SQLCA.SQLCODE = NOTFOUND THEN
               ERROR "No existen registros que cumplan la condicion...."
               LET win_flag = 0
               CLOSE WINDOW w_ropa205
               CLOSE ropa20_con
               FREE  ropa20_con
               RETURN FALSE
            ELSE
               DECLARE ropa202_prt CURSOR FOR ropa20_stmt2
               OPEN    ropa202_prt
               FETCH   ropa202_prt INTO ropa20_cnt
               CLOSE   ropa202_prt
               FREE    ropa202_prt
               CALL display_ropa20()
               MESSAGE "Existen ", ropa20_cnt USING "<<<<", " registros..."
               RETURN TRUE
            END IF
      END IF
   CLOSE WINDOW w_ropa205

END FUNCTION

FUNCTION apun20(fetch_flag)

   DEFINE fetch_flag SMALLINT
```

```
   FETCH RELATIVE fetch_flag ropa20_con INTO gi_ropa20.*

   IF SQLCA.SQLCODE = NOTFOUND THEN
      IF fetch_flag = 1 THEN
         ERROR "Esta ubicado al final de la tabla..."
      ELSE
         ERROR "Esta ubicado al principio de la tabla..."
      END IF
   ELSE
      CURRENT WINDOW is w_ropa205
      CALL display_ropa20()
   END IF

END FUNCTION

FUNCTION display_ropa20()

   DISPLAY BY NAME gi_ropa20.*

END FUNCTION

FUNCTION clean_ropa20()

   WHENEVER ERROR CONTINUE
   CLOSE ropa20_con
   FREE  ropa20_con
   WHENEVER ERROR STOP

END FUNCTION
```

```
-- ********************************************
-- * SISTEMA  : Uniformes y ropa de trabajo   *
-- * PROGRAMA : rop03m.4gl                    *
-- * OBJETIVO : Altas, bajas y cambios al     *
-- *            catalogo de categorias ropa20 *
-- * AUTOR    : Monica Ramirez Hernandez      *
-- * FECHA    : 07 de agosto de 1996          *
-- ********************************************

GLOBALS
   "rop01b.4gl"

FUNCTION rop03m()

   DEFINE alta CHAR(1)
   DEFINE baja CHAR(1)
   DEFINE  dr_aux record
   rp2_tnomi CHAR(1), rp2_adscr CHAR(8), rp2_categ CHAR(5), rp2_sexo CHAR(1)
   end record
   DEFINE tnomi CHAR(1), adscr CHAR(8), categ CHAR(5), sexo CHAR(1)

   DEFINE contador INTEGER

   INITIALIZE gr_ropa20.* TO NULL
   LET tnomi = NULL
   LET adscr = NULL
   LET categ = NULL
   LET sexo = NULL
   LET contador = 0
   OPEN FORM f03a FROM "rop03a"
   DISPLAY FORM f03a

      INPUT BY NAME dr_aux.*

        AFTER FIELD rp2_tnomi
          IF dr_aux.rp2_tnomi <> "1" and dr_aux.rp2_tnomi <> "2" and
             dr_aux.rp2_tnomi <> "3"  and dr_aux.rp2_tnomi <> "4" and
             dr_aux.rp2_tnomi <> "5" and dr_aux.rp2_tnomi <> "7" then
             error "Valor invalido. Verifique..."
             NEXT FIELD rp2_tnomi
          ELSE
             let tnomi = dr_aux.rp2_tnomi
          END IF

        AFTER FIELD rp2_adscr
        {   Select depstc from emadscr
           where depstc = dr_aux.rp2_adscr
           IF sqlca.sqlcode = 100 then
              error "Adscripcion no existente en catalogo, verifique..."
              NEXT FIELD rp2_adscr
           ELSE}
              let adscr = dr_aux.rp2_adscr
        --   END IF


        AFTER FIELD rp2_categ
        {  Select catstc from emcatego
           where catstc = dr_aux.rp2_categ
           IF sqlca.sqlcode = 100 then
              error "Categoria no existente en catalogo, verifique..."
              NEXT FIELD rp2_categ
           ELSE}
              let categ = dr_aux.rp2_categ
           --END IF

        AFTER FIELD rp2_sexo
          IF dr_aux.rp2_sexo <> "F" and dr_aux.rp2_sexo <> "M" then
             error "Valor invalido, Verifique..."
             NEXT FIELD rp2_sexo
          ELSE
             let sexo = dr_aux.rp2_sexo
          END IF

   END INPUT
   CLOSE FORM f03a

   OPEN FORM f03m FROM "rop03m"
   DISPLAY FORM f03m

   WHILE TRUE
      LET alta = NULL
      LET baja = NULL


      INPUT BY NAME gr_ropa20.*

        AFTER FIELD rp2_cverh1
          IF gr_ropa20.rp2_cverh1 <> "    " and
             gr_ropa20.rp2_cverh1 is not null then

          let gr_ropa20.rp2_cverh1 = gr_ropa20.rp2_cverh1 using "####"
          Select rp1_cverh from ropa10
          where rp1_cverh = gr_ropa20.rp2_cverh1
          IF sqlca.sqlcode = 100 then
             error "Clave no existente en catalogo, verifique..."
             NEXT FIELD rp2_cverh1
          END IF
          END IF

        AFTER FIELD rp2_cverh2
          IF gr_ropa20.rp2_cverh2 <> "    " and
             gr_ropa20.rp2_cverh2 is not null then
          let gr_ropa20.rp2_cverh2 = gr_ropa20.rp2_cverh2 using "####"
          Select rp1_cverh from ropa10
          where rp1_cverh = gr_ropa20.rp2_cverh2
          IF sqlca.sqlcode = 100 then
             error "Clave no existente en catalogo, verifique..."
             NEXT FIELD rp2_cverh2
          END IF
          END IF

        AFTER FIELD rp2_cverh3
          IF gr_ropa20.rp2_cverh3 <> "    " and
             gr_ropa20.rp2_cverh3 is not null then
```

```
    let gr_ropa20.rp2_cverh3 = gr_ropa20.rp2_cverh3 using "####"
    Select rp1_cverh from ropa10
    where rp1_cverh = gr_ropa20.rp2_cverh3
    IF sqlca.sqlcode = 100 then
       error "Clave no existente en catalogo, verifique..."
       NEXT FIELD rp2_cverh3
    END IF
    END IF

  AFTER FIELD rp2_cverh4
    IF gr_ropa20.rp2_cverh4 <> "    " and
       gr_ropa20.rp2_cverh4 is not null then
    let gr_ropa20.rp2_cverh4 = gr_ropa20.rp2_cverh4 using "####"
    Select rp1_cverh from ropa10
    where rp1_cverh = gr_ropa20.rp2_cverh4
    IF sqlca.sqlcode = 100 then
       error "Clave no existente en catalogo, verifique..."
       NEXT FIELD rp2_cverh4
    END IF
    END IF

  AFTER FIELD rp2_cverh5
    IF gr_ropa20.rp2_cverh5 <> "    " and
       gr_ropa20.rp2_cverh5 is not null then
    let gr_ropa20.rp2_cverh5 = gr_ropa20.rp2_cverh5 using "####"
    Select rp1_cverh from ropa10
    where rp1_cverh = gr_ropa20.rp2_cverh5
    IF sqlca.sqlcode = 100 then
       error "Clave no existente en catalogo, verifique .."
       NEXT FIELD rp2_cverh5
    END IF
    END IF

  AFTER FIELD rp2_cverh6
    IF gr_ropa20.rp2_cverh6 <> "    " and
       gr_ropa20.rp2_cverh6 is not null then
    let gr_ropa20.rp2_cverh6 = gr_ropa20.rp2_cverh6 using "####"
    Select rp1_cverh from ropa10
    where rp1_cverh = gr_ropa20.rp2_cverh6
    IF sqlca.sqlcode = 100 then
       error "Clave no existente en catalogo, verifique .."
       NEXT FIELD rp2_cverh6
    END IF
    END IF

  AFTER FIELD rp2_cverh7
    IF gr_ropa20.rp2_cverh7 <> "    " and
       gr_ropa20.rp2_cverh7 is not null then
    let gr_ropa20.rp2_cverh7 = gr_ropa20.rp2_cverh7 using "####"
    Select rp1_cverh from ropa10
    where rp1_cverh = gr_ropa20.rp2_cverh7
    IF sqlca.sqlcode = 100 then
       error "Clave no existente en catalogo, verifique..."
       NEXT FIELD rp2_cverh7
    END IF
    END IF

  AFTER FIELD rp2_cverh8
    IF gr_ropa20.rp2_cverh8 <> "    " and
       gr_ropa20.rp2_cverh8 is not null then
    let gr_ropa20.rp2_cverh8 = gr_ropa20.rp2_cverh8 using "####"
    Select rp1_cverh from ropa10
    where rp1_cverh = gr_ropa20.rp2_cverh8
    IF sqlca.sqlcode = 100 then
       error "Clave no existente en catalogo, verifique..."
       NEXT FIELD rp2_cverh8
    END IF
    END IF

  AFTER FIELD rp2_cverh9
    IF gr_ropa20.rp2_cverh9 <> "    " and
       gr_ropa20.rp2_cverh9 is not null then
    let gr_ropa20.rp2_cverh9 = gr_ropa20.rp2_cverh9 using "####"
    Select rp1_cverh from ropa10
    where rp1_cverh = gr_ropa20.rp2_cverh9
    IF sqlca.sqlcode = 100 then
       error "Clave no existente en catalogo, verifique.. "
       NEXT FIELD rp2_cverh9
    END IF
    END IF

  AFTER FIELD rp2_cverh10
    IF gr_ropa20.rp2_cverh10 <> "    " and
       gr_ropa20.rp2_cverh10 is not null then
    let gr_ropa20.rp2_cverh10 = gr_ropa20.rp2_cverh10 using "####"
    Select rp1_cverh from ropa10
    where rp1_cverh = gr_ropa20.rp2_cverh10
    IF sqlca.sqlcode = 100 then
       error "Clave no existente en catalogo, verifique..."
       NEXT FIELD rp2_cverh10
    END IF
    END IF

  AFTER FIELD rp2_cverh11
    IF gr_ropa20.rp2_cverh11 <> "    " and
       gr_ropa20.rp2_cverh11 is not null then
    let gr_ropa20.rp2_cverh11 = gr_ropa20.rp2_cverh11 using "####"
    Select rp1_cverh from ropa10
    where rp1_cverh = gr_ropa20.rp2_cverh11
    IF sqlca.sqlcode = 100 then
       error "Clave no existente en catalogo, verifique..."
       NEXT FIELD rp2_cverh11
    END IF
    END IF

  AFTER FIELD rp2_cverh12
    IF gr_ropa20.rp2_cverh12 <> "    " and
       gr_ropa20.rp2_cverh12 is not null then
    let gr_ropa20.rp2_cverh12 = gr_ropa20.rp2_cverh12 using "####"
    Select rp1_cverh from ropa10
    where rp1_cverh = gr_ropa20.rp2_cverh12
    IF sqlca.sqlcode = 100 then
       error "Clave no existente en catalogo, verifique..."
       NEXT FIELD rp2_cverh12
```

```
            END IF
            END IF

         AFTER FIELD rp2_cverh13
            IF gr_ropa20.rp2_cverh13 <> "    " and
               gr_ropa20.rp2_cverh13 is not null then
            let gr_ropa20.rp2_cverh13 = gr_ropa20.rp2_cverh13 using "####"
            Select rp1_cverh from ropa10
            where rp1_cverh = gr_ropa20.rp2_cverh13
            if sqlca.sqlcode = 100 then
               error "Clave no existente en catalogo, verifique .."
               NEXT FIELD rp2_cverh13
            END IF
            END IF

         AFTER FIELD rp2_cverh14
            IF gr_ropa20.rp2_cverh14 <> "    " and
               gr_ropa20.rp2_cverh14 is not null then
            let gr_ropa20.rp2_cverh14 = gr_ropa20.rp2_cverh14 using "####"
            Select rp1_cverh from ropa10
            where rp1_cverh = gr_ropa20.rp2_cverh14
            if sqlca.sqlcode = 100 then
               error "Clave no existente en catalogo, verifique. ."
               NEXT FIELD rp2_cverh14
            END IF
            END IF

         AFTER FIELD rp2_cverh15
            IF gr_ropa20.rp2_cverh15 <> "    " and
               gr_ropa20.rp2_cverh15 is not null then
            let gr_ropa20.rp2_cverh15 = gr_ropa20.rp2_cverh15 using "####"
            Select rp1_cverh from ropa10
            where rp1_cverh = gr_ropa20.rp2_cverh15
            if sqlca.sqlcode = 100 then
               error "Clave no existente en catalogo, verifique  ."
               NEXT FIELD rp2_cverh15
            END IF
            END IF

         KEYDOWN FIELD rp2_ident

             SELECT rp2_ident,rp2_cverh1,rp2_cant1), rp2_cverh2,rp2_cant12,
                    rp2_cverh1,rp2_cant1), rp2_cverh4,rp2_cant14,
                    rp2_cverh5,rp2_cant15, rp2_cverh6,rp2_cant16,
                    rp2_cverh7,rp2_cant17, rp2_cverh8,rp2_cant18,
                    rp2_cverh9,rp2_cant19, rp2_cverh10,rp2_cant110,
                    rp2_cverh11,rp2_cant111, rp2_cverh12,rp2_cant112,
                    rp2_cverh13,rp2_cant113, rp2_cverh14,rp2_cant114,
                    rp2_cverh15,rp2_cant115
             INTO   gr_ropa20.*
             FROM   ropa20
             WHERE  ropa20.rp2_tnomi = tnomi AND
                    ropa20.rp2_adscr = adscr AND
                    ropa20.rp2_categ = categ AND
                    ropa20.rp2_sexo  = sexo

             IF SQLCA.SQLCODE = 100 THEN
                MESSAGE "TIPO DE NOMINA : ",tnomi," ADSCRIPCION : ", adscr, "  CATEGORIA : ",categ," SEXO : ",sexo, " "
                OPEN WINDOW pide AT 12,10 WITH 3 ROWS, 50 COLUMNS ATTRIBUTE(BORDER)
                   PROMPT "Registro no Existente, Desea dar de Alta (S/N) ? " FOR alta
                CLOSE WINDOW pide

                IF UPSHIFT(alta) = "N" THEN
                   CLEAR FORM
                   INITIALIZE gr_ropa20 * TO NULL
                   LET tnomi = NULL
                   LET adscr = NULL
                   LET categ = NULL
                   LET sexo = NULL
                   CLEAR SCREEN
                   RETURN
                END IF
             ELSE
                MESSAGE "TIPO DE NOMINA : ",tnomi," ADSCRIPCION : ", adscr, "  CATEGORIA : ",categ," SEXO : ",sexo, " "

                DISPLAY BY NAME gr_ropa20.*

             END IF

             ON KEY (CONTROL-B)

             OPEN WINDOW pibe AT 12,15 WITH 3 ROWS, 40 COLUMNS ATTRIBUTE(BORDER)
                PROMPT "Confirma la baja del Registro (S/N) ? " FOR baja
             CLOSE WINDOW pibe
             IF UPSHIFT(baja) = "S" THEN
                DELETE FROM ropa20
                WHERE  ropa20.rp2_tnomi = tnomi AND
                       ropa20.rp2_adscr = adscr AND
                       ropa20.rp2_categ = categ AND
                       ropa20.rp2_sexo  = sexo

                INSERT INTO ropbis20 VALUES(tnomi,adscr,categ,sexo,current,
                                            contador,"B",gr_ropa20.*)

                ERROR "BAJA REALIZADA CON EXITO ..."
                SLEEP 3
                INITIALIZE gr_ropa20.* TO NULL
                LET tnomi = NULL
                LET adscr = NULL
                LET categ = NULL
                LET sexo = NULL
                CLEAR SCREEN
                RETURN
             ELSE
                INITIALIZE gr_ropa20.* TO NULL
                LET tnomi = NULL
                LET adscr = NULL
                LET categ = NULL
                LET sexo = NULL
                CLEAR SCREEN
                RETURN
             END IF
END INPUT
```

```
IF INT_FLAG = TRUE THEN
   LET INT_FLAG = FALSE
   CLEAR FORM
   ERROR "PROCESO CANCELADO ..."
   SLEEP 2
   CLEAR SCREEN
   EXIT WHILE
   RETURN
ELSE
   IF UPSHIFT(ela) = "A" THEN
      IF gr_ropa20.rp2_cverh1 <> "    " or
         gr_ropa20.rp2_cverh1 is not null THEN
         let contador = contador + 1
      END IF
      IF gr_ropa20.rp2_cverh2 <> "    " or
         gr_ropa20.rp2_cverh2 is not null THEN
         let contador = contador + 1
      END IF
      IF gr_ropa20.rp2_cverh3 <> "    " or
         gr_ropa20.rp2_cverh3 is not null THEN
         let contador = contador + 1
      END IF
      IF gr_ropa20.rp2_cverh4 <> "    " or
         gr_ropa20.rp2_cverh4 is not null THEN
         let contador = contador + 1
      END IF
      IF gr_ropa20.rp2_cverh5 <> "    " or
         gr_ropa20.rp2_cverh5 is not null THEN
         let contador = contador + 1
      END IF
      IF gr_ropa20.rp2_cverh6 <> "    " or
         gr_ropa20.rp2_cverh6 is not null THEN
         let contador = contador + 1
      END IF
      IF gr_ropa20.rp2_cverh7 <> "    " or
         gr_ropa20.rp2_cverh7 is not null THEN
         let contador = contador + 1
      END IF
      IF gr_ropa20.rp2_cverh8 <> "    " or
         gr_ropa20.rp2_cverh8 is not null THEN
         let contador = contador + 1
      END IF
      IF gr_ropa20.rp2_cverh9 <> "    " or
         gr_ropa20.rp2_cverh9 is not null THEN
         let contador = contador + 1
      END IF
      IF gr_ropa20.rp2_cverh10 <> "    " or
         gr_ropa20.rp2_cverh10 is not null THEN
         let contador = contador + 1
      END IF
      IF gr_ropa20.rp2_cverh11 <> "    " or
         gr_ropa20.rp2_cverh11 is not null THEN
         let contador = contador + 1
      END IF
      IF gr_ropa20.rp2_cverh12 <> "    " or
         gr_ropa20.rp2_cverh12 is not null THEN
         let contador = contador + 1
      END IF
      IF gr_ropa20.rp2_cverh13 <> "    " or
         gr_ropa20.rp2_cverh13 is not null THEN
         let contador = contador + 1
      END IF
      IF gr_ropa20.rp2_cverh14 <> "    " or
         gr_ropa20.rp2_cverh14 is not null THEN
         let contador = contador + 1
      END IF
      IF gr_ropa20.rp2_cverh15 <> "    " or
         gr_ropa20.rp2_cverh15 is not null THEN
         let contador = contador + 1
      END IF

         let gr_ropa20.rp2_cverh1 = gr_ropa20.rp2_cverh1 using "####"
         let gr_ropa20.rp2_cant11 = gr_ropa20.rp2_cant11 using "##"
         let gr_ropa20.rp2_cverh2 = gr_ropa20.rp2_cverh2 using "####"
         let gr_ropa20.rp2_cant12 = gr_ropa20.rp2_cant12 using "##"
         let gr_ropa20.rp2_cverh3 = gr_ropa20.rp2_cverh3 using "####"
         let gr_ropa20.rp2_cant13 = gr_ropa20.rp2_cant13 using "##"
         let gr_ropa20.rp2_cverh4 = gr_ropa20.rp2_cverh4 using "####"
         let gr_ropa20.rp2_cant14 = gr_ropa20.rp2_cant14 using "##"
         let gr_ropa20.rp2_cverh5 = gr_ropa20.rp2_cverh5 using "####"
         let gr_ropa20.rp2_cant15 = gr_ropa20.rp2_cant15 using "##"
         let gr_ropa20.rp2_cverh6 = gr_ropa20.rp2_cverh6 using "####"
         let gr_ropa20.rp2_cant16 = gr_ropa20.rp2_cant16 using "##"
         let gr_ropa20.rp2_cverh7 = gr_ropa20.rp2_cverh7 using "####"
         let gr_ropa20.rp2_cant17 = gr_ropa20.rp2_cant17 using "##"
         let gr_ropa20.rp2_cverh8 = gr_ropa20.rp2_cverh8 using "####"
         let gr_ropa20.rp2_cant18 = gr_ropa20.rp2_cant18 using "##"
         let gr_ropa20.rp2_cverh9 = gr_ropa20.rp2_cverh9 using "####"
         let gr_ropa20.rp2_cant19 = gr_ropa20.rp2_cant19 using "##"
         let gr_ropa20.rp2_cverh10 = gr_ropa20.rp2_cverh10 using "####"
         let gr_ropa20.rp2_cant110 = gr_ropa20.rp2_cant110 using "##"
         let gr_ropa20.rp2_cverh11 = gr_ropa20.rp2_cverh11 using "####"
         let gr_ropa20.rp2_cant111 = gr_ropa20.rp2_cant111 using "##"
         let gr_ropa20.rp2_cverh12 = gr_ropa20.rp2_cverh12 using "####"
         let gr_ropa20.rp2_cant112 = gr_ropa20.rp2_cant112 using "##"
         let gr_ropa20.rp2_cverh13 = gr_ropa20.rp2_cverh13 using "####"
         let gr_ropa20.rp2_cant113 = gr_ropa20.rp2_cant113 using "##"
         let gr_ropa20.rp2_cverh14 = gr_ropa20.rp2_cverh14 using "####"
         let gr_ropa20.rp2_cant114 = gr_ropa20.rp2_cant114 using "##"
         let gr_ropa20.rp2_cverh15 = gr_ropa20.rp2_cverh15 using "####"
         let gr_ropa20.rp2_cant115 = gr_ropa20.rp2_cant115 using "##"

      INSERT INTO ropa20 VALUES(nomi,adscr,categ,sexo,contador,
                                gr_ropa20.*)

      INSERT INTO ropha20 VALUES(nomi,adscr,categ,sexo,current,
                                 contador,"A",gr_ropa20.*)

      let contador = 0
      ERROR "REGISTRO AGREGADO CON EXITO . "
      SLEEP 3
      CLEAR SCREEN
      RETURN
   ELSE
```

```
   IF gr_ropa20.rp2_cverh1 <> "    " or
      gr_ropa20.rp2_cverh1 is not null THEN
      let contador = contador + 1
   END IF
   IF gr_ropa20.rp2_cverh2 <> "    " or
      gr_ropa20.rp2_cverh2 is not null THEN
      let contador = contador + 1
   END IF
   IF gr_ropa20.rp2_cverh3 <> "    " or
      gr_ropa20.rp2_cverh3 is not null THEN
      let contador = contador + 1
   END IF
   IF gr_ropa20.rp2_cverh4 <> "    " or
      gr_ropa20.rp2_cverh4 is not null THEN
      let contador = contador + 1
   END IF
   IF gr_ropa20.rp2_cverh5 <> "    " or
      gr_ropa20.rp2_cverh5 is not null THEN
      let contador = contador + 1
   END IF
   IF gr_ropa20.rp2_cverh6 <> "    " or
      gr_ropa20.rp2_cverh6 is not null THEN
      let contador = contador + 1
   END IF
   IF gr_ropa20.rp2_cverh7 <> "    " or
      gr_ropa20.rp2_cverh7 is not null THEN
      let contador = contador + 1
   END IF
   IF gr_ropa20.rp2_cverh8 <> "    " or
      gr_ropa20.rp2_cverh8 is not null THEN
      let contador = contador + 1
   END IF
   IF gr_ropa20 rp2_cverh9 <> "    " or
      gr_ropa20.rp2_cverh9 is not null THEN
      let contador = contador + 1
   END IF
   IF gr_ropa20.rp2_cverh10 <> "    " or
      gr_ropa20.rp2_cverh10 is not null THEN
      let contador = contador + 1
   END IF
   IF gr_ropa20.rp2_cverh11 <> "    " or
      gr_ropa20.rp2_cverh11 is not null THEN
      let contador = contador + 1
   END IF
   IF gr_ropa20.rp2_cverh12 <> "    " or
      gr_ropa20.rp2_cverh12 is not null THEN
      let contador = contador + 1
   END IF
   IF gr_ropa20.rp2_cverh13 <> "    " or
      gr_ropa20.rp2_cverh13 is not null THEN
      let contador = contador + 1
   END IF
   IF gr_ropa20.rp2_cverh14 <> "    " or
      gr_ropa20.rp2_cverh14 is not null THEN
      let contador = contador + 1
   END IF
   IF gr_ropa20.rp2_cverh15 <> "    " or
      gr_ropa20.rp2_cverh15 is not null THEN
      let contador = contador + 1
   END IF
      let gr_ropa20.rp2_cverh1 = gr_ropa20.rp2_cverh1 using "####"
      let gr_ropa20.rp2_canti1 = gr_ropa20.rp2_canti1 using "##"
      let gr_ropa20.rp2_cverh2 = gr_ropa20.rp2_cverh2 using "####"
      let gr_ropa20.rp2_canti2 = gr_ropa20.rp2_canti2 using "##"
      let gr_ropa20.rp2_cverh3 = gr_ropa20.rp2_cverh3 using "####"
      let gr_ropa20.rp2_canti3 = gr_ropa20.rp2_canti3 using "##"
      let gr_ropa20.rp2_cverh4 = gr_ropa20.rp2_cverh4 using "####"
      let gr_ropa20.rp2_canti4 = gr_ropa20.rp2_canti4 using "##"
      let gr_ropa20.rp2_cverh5 = gr_ropa20.rp2_cverh5 using "####"
      let gr_ropa20.rp2_canti5 = gr_ropa20.rp2_canti5 using "##"
      let gr_ropa20.rp2_cverh6 = gr_ropa20.rp2_cverh6 using "####"
      let gr_ropa20.rp2_canti6 = gr_ropa20.rp2_canti6 using "##"
      let gr_ropa20.rp2_cverh7 = gr_ropa20.rp2_cverh7 using "####"
      let gr_ropa20.rp2_canti7 = gr_ropa20.rp2_canti7 using "##"
      let gr_ropa20.rp2_cverh8 = gr_ropa20.rp2_cverh8 using "####"
      let gr_ropa20.rp2_canti8 = gr_ropa20.rp2_canti8 using "##"
      let gr_ropa20.rp2_cverh9 = gr_ropa20.rp2_cverh9 using "####"
      let gr_ropa28.rp2_canti9 = gr_ropa30.rp2_canti9 using "##"
      let gr_ropa20.rp2_cverh10 = gr_ropa20.rp2_cverh10 using "####"
      let gr_ropa20.rp2_canti10 = gr_ropa20.rp2_canti10 using "##"
      let gr_ropa20.rp2_cverh11 = gr_ropa20.rp2_cverh11 using "####"
      let gr_ropa20.rp2_canti11 = gr_ropa20.rp2_canti11 using "##"
      let gr_ropa20.rp2_cverh12 = gr_ropa20.rp2_cverh12 using "####"
      let gr_ropa20.rp2_canti12 = gr_ropa20.rp2_canti12 using "##"
      let gr_ropa20.rp2_cverh13 = gr_ropa20.rp2_cverh13 using "####"
      let gr_ropa20.rp2_canti13 = gr_ropa20.rp2_canti13 using "##"
      let gr_ropa20.rp2_cverh14 = gr_ropa20.rp2_cverh14 using "####"
      let gr_ropa20.rp2_canti14 = gr_ropa20.rp2_canti14 using "##"
      let gr_ropa20.rp2_cverh15 = gr_ropa20.rp2_cverh15 using "####"
      let gr_ropa20.rp2_canti15 = gr_ropa20.rp2_canti15 using "##"
   UPDATE ropa20 SET (rp2_tnom1,rp2_adscr,rp2_categ,rp2_sexo,
                      rp2_cont,rp2_ident,
                      rp2_cverh1,rp2_canti1, rp2_cverh2,rp2_canti2,
                      rp2_cverh3,rp2_canti3, rp2_cverh4,rp2_canti4,
                      rp2_cverh5,rp2_canti5, rp2_cverh6,rp2_canti6,
                      rp2_cverh7,rp2_canti7, rp2_cverh8,rp2_canti8,
                      rp2_cverh9,rp2_canti9, rp2_cverh10,rp2_canti10,
                      rp2_cverh11,rp2_canti11, rp2_cverh12,rp2_canti12,
                      rp2_cverh13,rp2_canti13, rp2_cverh14,rp2_canti14,
                      rp2_cverh15,rp2_canti15) = (tnom1,adscr,categ,
                      sexo,contador,gr_ropa20.*)
   WHERE ropa20.rp2_tnom1 = tnom1 AND
         ropa20.rp2_adscr = adscr AND
         ropa20.rp2_categ = categ AND
         ropa20.rp2_sexo  = sexo

   INSERT INTO ropbig20 VALUES(tnom1,adscr,categ,sexo,current,
                               contador,"C",gr_ropa20.*)

   let contador = 0
   ERROR "CAMBIO REALIZADO CON EXITO .. "
   SLEEP 3
   CLEAR SCREEN
   RETURN
```

```
        END IF
      END IF

      IF SQLCA.SQLCODE < 0 THEN
        ERROR "PROBLEMAS EN LA ACTUALIZACION ..."
      END IF
    END WHILE
END FUNCTION
```

```
-- *****************************************
-- *  SISTEMA  : Uniformes y ropa de trabajo *
-- *  PROGRAMA : rop04r.4gl                  *
   *  OBJETIVO : Genera un reporte del catalogo *
   *             de categorias ropa20       *
-- *  AUTOR    : Monica Ramirez Hernandez    *
-- *  FECHA    : 01 de agosto de 1994        *
   *****************************************

database sirh

FUNCTION rop04r()
define vm_numcma    smallint
define vm_numcmachr char(2)
define vm_nomarch   varchar(20)
define vm_nombre    varchar(100)
define vm_comando   varchar(90)
            MENU "Enviar Reporte a "
                    COMMAND "Impresora" "Envia Reporte a la Impresora"
                            call crea_nomarch("rop04r") returning vm_nomarch
                               call imp_imp(10,10,vm_nomarch) returning vm_comando
                               START REPORT rep_04r TO vm_nomarch
                               CALL rop04ra()
                               call manda_impre(vm_comando,vm_nomarch)
                               call borra_arch(vm_nomarch)
                    COMMAND "Archivo" "Envia Reporte a un Archivo"
                            call pide_nombre(10,10) returning vm_nombre
                            if vm_nombre is null then
                                    next option "Salida"
                            else
                               START REPORT rep_04r TO vm_nombre
                               CALL rop04ra()
                               next option "Reimpresion"
                            end if
                    COMMAND "Pantalla" "Envia Reporte a la Pantalla"
                               call crea_nomarch("rop04r") returning vm_nomarch
                               START REPORT rep_04r TO vm_nomarch
                               CALL rop04ra()
                               call mata_archivo(vm_nomarch)
                    COMMAND "Reimpresion" "Envia a impresora el reporte generado en la opcion <Archivo>"
                               call pide_nombre(10,10) returning vm_nombre
                               if vm_nombre is null then
                                       next option "Salida"
                               else
                                  call imp_imp(10,10,vm_nombre) returning vm_comando
                                  call manda_impre(vm_comando,vm_nombre)
                                  call borra_arch(vm_nombre)
                               end if
                    COMMAND "Salida" "Salir al Menu Principal"
                               EXIT MENU
        END MENU
END FUNCTION

function rop04ra()
        define tr_04r record  like ropa30.*
        define orden    char(1),
               genero   char(1),
               ident    char(1),
               consult  char(350),
               consulta char(350),
               ordena   char(90),
               nu_repor char(35),
        open form for04 from "rop04r"
        display form  for04
               let int_flag = false
               input orden from orden
               after field orden
                     if orden is null and
                        orden < 1     and
                        orden > 9     then
                        error "VALOR INVALIDO,VERIFIQUE..."
                        next field orden
                     end if
               on key (control-w)
               if infield(orden) then
                     call showhelp(150)
               end if
               end input
               if int_flag = true then
                     let int_flag = false
                     initialize orden to null
                     error "REPORTE CANCELADO"
                     sleep 2
                     clear screen
                     close form for04
                     return
               end if
                     close form for04
               let consulta = "select *,rp2_adscr[1,3] from ropa20 "
               case
                  when orden = "1"
                     let ordena = "order by 2,1,3,4"
                  when orden = "2"
                     let ordena = "order by 3"
                  when orden = "3"
                     let ordena = "order by 2,3"
                  when orden = "4"
                     let ordena = "order by 2,6,4"
                  when orden = "5"
                     let ordena = "order by 3"
                  when orden = "6"
                     let ordena = "order by 4"
                  when orden = "7"
                     let ordena = "order by 4,1"
                  when orden = "8"
                     let ordena = "order by 2,1"
               end case
               {let nu_repor = "/usera/produ/reportes/rop04r".orden clipped,".rep"}
               let consult = consulta,ordena
               prepare tr_04 from consult
               declare c_rop04r cursor for tr_04
               error  "Favor de esperar ..Gracias"
           {start report rep_04r to nu_repor
           error "generando reporte /usera/produ/reportes/rop04r".orden,".rep"}
```

```
    foreach cur_rep04r into lr_04r.*,geren
            output to report rep_04r(lr_04r.*,geren,orden)
    end foreach
        finish report rep_04r
        clear screen
end function

report rep_04r(r04r,rgeren,orden)
define r04r record like ropa20.*
define orden char(1)
define rgeren char(3)
define var char(10)
{
output
    left margin 0
}
format
page header
{case
    when r04r.rp2_tnomi = 1
        let var = "BASE"
    when r04r.rp2_tnomi = 2
        let var = "CONFIANZA"
    when r04r.rp2_tnomi = 3
        let var = "EVENTUAL"
end case}
     print column   1, "rep04r",orden,".rep",
           column  90, "SISTEMA DE TRANSPORTE COLECTIVO (METRO)",
           column 194, "FECHA : ",today
     print column  94, "UNIFORMES Y ROPA DE TRABAJO",
           column 165, "GERENCIA DE RECURSOS HUMANOS",
           column 194, "PAG.  : ", pageno using "##,###,###"
     print column  81, "RELACION DEL CATALOGO DE ASIGNACION DE ROPA POR CATEGORIA",
           column 194, "HORA  :    ",time
     print "--------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------"
    --print column   1, "TIPO DE NOMINA : ", r04r.rp2_tnomi," ",var
    --print "--------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------"
     print column   1, "ADSCRIP.",
           column  10, "CAT ",
           column  16, "S",
           column  18, "I",
           column  20, "N",
           column  22, "CVE1",
           column  27, "CANT1",
           column  33, "CVE2",
           column  39, "CANT2",
           column  45, "CVE3",
           column  51, "CANT3",
           column  57, "CVE4",
           column  63, "CANT4",
           column  69, "CVE5",
           column  75, "CANT5",
           column  81, "CVE6",
           column  87, "CANT6",
           column  93, "CVE7",
           column  99, "CANT7",
           column 105, "CVE8",
           column 111, "CANT8",
           column 117, "CVE9",
           column 123, "CANT9",
           column 129, "CVE10",
           column 136, "CANT10",
           column 143, "CVE11",
           column 150, "CANT11",
           column 157, "CVE12",
           column 164, "CANT12",
           column 171, "CVE13",
           column 178, "CANT13",
           column 185, "CVE14",
           column 192, "CANT14",
           column 199, "CVE15",
           column 206, "CANT15"
     print "--------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------"

{before group of r04r.rp2_tnomi
      skip to top of page}

on every row
     print column   1, r04r.rp2_adscr,
           column  10, r04r.rp2_categ,
           column  16, r04r.rp2_sexo,
           column  18, r04r.rp2_ident,
           column  20, r04r.rp2_tnomi,
           column  22, r04r.rp2_cverh1,
           column  27, r04r.rp2_canti1,
           column  33, r04r.rp2_cverh2,
           column  39, r04r.rp2_canti2,
           column  45, r04r.rp2_cverh3,
           column  51, r04r.rp2_canti3,
           column  57, r04r.rp2_cverh4,
           column  63, r04r.rp2_canti4,
           column  69, r04r.rp2_cverh5,
           column  75, r04r.rp2_canti5,
           column  81, r04r.rp2_cverh6,
           column  87, r04r.rp2_canti6,
           column  93, r04r.rp2_cverh7,
           column  99, r04r.rp2_canti7,
           column 105, r04r.rp2_cverh8,
           column 111, r04r.rp2_canti8,
           column 117, r04r.rp2_cverh9,
           column 123, r04r.rp2_canti9,
           column 129, r04r.rp2_cverh10,
           column 136, r04r.rp2_canti10,
           column 143, r04r.rp2_cverh11,
           column 150, r04r.rp2_canti11,
           column 157, r04r.rp2_cverh12,
           column 164, r04r.rp2_canti12,
           column 171, r04r.rp2_cverh13,
           column 178, r04r.rp2_canti13,
           column 185, r04r.rp2_cverh14,
           column 192, r04r.rp2_canti14,
```

```
          column 195, r04r.rp2_dvarh15,
          column 204, r04r.rp2_cant15
after group of r04r.rp2_cnom4
     skip 1 line
      print column 1, "TOTAL POR TIPO DE NOMINA : ",group count (*) using "####"
on last row
     skip 1 line
      print column 9, "TOTAL GENERAL  : ", count(*) using "##,###,###"
end report
```

```
-- ****************************************
-- * SISTEMA   : Uniformes y ropa de trabajo *
-- * PROGRAMA  : rop05r.4gl                  *
-- * OBJETIVO  : Reporte de trabajadores     *
-- *             por categoria calificada    *
-- * AUTOR     : Monica Ramirez Hernandez    *
-- * FECHA     : 01 de agosto de 1996        *
-- ****************************************

database simh
        globals
                define r_var record  like ropa20.*
                define cantidad char(4)
        end globals

FUNCTION rop05r()
define vm_numcna    smallint
define vm_numcnachr char(2)
define vm_nomarch   varchar(20)
define vm_nombre    varchar(100)
define vm_comando   varchar(50)
                MENU "Enviar Reporte a "
                        COMMAND "Impresora" "Envia Reporte a la Impresora"
                                call crea_nomarch("rop05r") returning vm_nomarch
                                    call imp_imp(10,10,vm_nomarch) returning vm_comando
                                    START REPORT rep_05r TO vm_nomarch
                                    CALL rop05ra()
                                    call manda_impre(vm_comando,vm_nomarch)
                                    call borra_arch(vm_nomarch)
                        COMMAND "Archivo" "Envia Reporte a un Archivo"
                                    call pide_nombre(10,10) returning vm_nombre
                                    if vm_nombre is null then
                                            next option "Salida"
                                    else
                                ERROR "Favor de Esperar  ..Gracias"
                                    START REPORT rep_05r TO vm_nombre
                                    CALL rop05ra()
                                    next option "Reimpresion"
                                end if
                        COMMAND "Pantalla" "Envia Reporte a la Pantalla"
                          ERROR "Favor de Esperar...Gracias"
                                    call crea_nomarch("rop05r") returning vm_nomarch
                                    START REPORT rep_05r TO vm_nomarch
                                    CALL rop05ra()
                                    call more_archivo(vm_nomarch)
                        COMMAND "Reimpresion" "Envia a impresora el reporte generado en la opcion <Archivo>"
                                    call pide_nombre(10,10) returning vm_nombre
                                    if vm_nombre is null then
                                            next option "Salida"
                                    else
                                      call imp_imp(10,10,vm_nombre) returning vm_comando
                                      call manda_impre(vm_comando,vm_nombre)
                                      call borra_arch(vm_nombre)
                                    end if
                        COMMAND "Salida" "Salir al Menu Principal"
                                    EXIT MENU
        END MENU
END FUNCTION

function rop05ra()
        declare cur_rop05r cursor for
                select count(*),r.*
                from   ropa20 r,empleado e
                where  r rp2_tnomi = e.tipnom
                and    r rp2_adscr = e depsto
                and    r.rp2_categ = e catsto
                and    r.rp2_sexo  = e.sexo
                and    e.idbaja <> '1'
                and    e.tiplic = '0'

                group by 2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,
                         22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37
        {start report rep_05r to "/users/produ/reportes/rop05r.rep"
            error "generando reporte /users/produ/reportes/rop05r.rep"}
        foreach cur_rop05r into cantidad,r_05r.*
            output to report rep_05r(cantidad,r_05r *)
        end foreach
                finish report rep_05r
                {   error "COMPRUEBE SU IMPRESORA PARA LETRA CHICA...GRACIAS"
                sleep 3}
                clear screen
end function
report rep_05r(cantidad,r05r)
        define r05r record like ropa20.*
        define cantidad  char(4),
                ident_r   smallint,
                ident_u   smallint,
                ident_e   smallint,
                total_id  smallint,
                catid_r   smallint,
                catid_u   smallint,
                catid_e   smallint,
                totals_id smallint,
                totgrl_r  smallint,
                totgrl_u  smallint,
                totgrl_e  smallint,
                totgrl_id smallint,
                mujeres   smallint,
                hombres   smallint,
                total     smallint,
                totmuj    smallint
                tothom    smallint,
                totales   smallint,
                totgrlmuj smallint,
                totgrlhom smallint,
                totalgnrl smallint

{
output
    left margin 0
}
order by r05r.rp2_tnomi,r05r.rp2_adscr,r05r.rp1_sexo,r05r.rp2_ident,
         r05r.rp2_categ
format
```

```
page header
        print column   1, "rop05r.rep",
              column  40, "SISTEMA DE TRANSPORTE COLECTIVO (METRO)",
              column 104, "FECHA : ",today
        print column   1, "DIRECCION DE ADMINISTRACION",
              column  94, "UNIFORMES Y ROPA DE TRABAJO",
              column 163, "GERENCIA DE RECURSOS HUMANOS",
              column 194, "PAG.  : ", pageno using "##,###,###"
        print column  82, "NUMERO DE TRABAJADORES POR CATEGORIA CALIFICADA".
              column 194, "HORA  :  ",time
        print '-------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------"

        print column   1, "T.NOMINA",
              column  15, "ADSCRIPCION",
              column  38, "CATEGORIA",
              column  63, "SEXO",
              column  50, "ID",
              column  55, "CANTIDAD",
              column 101, "C L A V E S   Y   C A N T I D A D E S"
        print '-------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------"

before group of r05r.rp2_tnomi
        skip to top of page
        let totmuj = 0
        let tothom = 0
        let totales = 0
        let totid_r  = 0
        let totid_u  = 0
        let totid_a  = 0
before group of r05r.rp2_adscr
        let mujeres = 0
        let hombres = 0
        let total  = 0
        let ident_r = 0
        let ident_u = 0
        let ident_a = 0
on every row
        skip 1 line
        print column   5, r05r.rp2_tnomi,
              column  17, r05r.rp2_adscr,
              column  32, r05r.rp2_categ,
              column  44, r05r.rp2_sexo,
              column  50, r05r.rp2_ident,
              column  54, cantidad,
              column  65, r05r.rp2_everh1,
              column  79, r05r.rp2_cant1,
              column  83, r05r.rp2_everh2,
              column  93, r05r.rp2_cant12,
              column 101, r05r.rp2_everh3,
              column 111, r05r.rp2_cant13,
              column 119, r05r.rp2_everh4,
              column 129, r05r.rp2_cant14,
              column 137, r05r.rp2_everh5,
              column 147, r05r.rp2_cant15,
              column 156, r05r.rp2_everh6,
              column 165, r05r.rp2_cant16,
              column 173, r05r.rp2_everh7,
              column 182, r05r.rp2_cant17,
              column 191, r05r.rp2_everh8,
              column 201, r05r.rp2_cant18
        print column  68, r05r.rp2_everh9,
              column  75, r05r.rp2_cant19,
              column  83, r05r.rp2_everh10,
              column  93, r05r.rp2_cant110,
              column 101, r05r.rp2_everh11,
              column 111, r05r.rp2_cant111,
              column 119, r05r.rp2_everh12,
              column 129, r05r.rp2_cant112,
              column 137, r05r.rp2_everh13,
              column 147, r05r.rp2_cant113,
              column 155, r05r.rp2_everh14,
              column 165, r05r.rp2_cant114,
              column 173, r05r.rp2_everh15,
              column 183, r05r.rp2_cant115
        case
              when
                   r05r.rp2_ident = "R"
                        let ident_r  = ident_r + cantidad
                        let totid_r  = totid_r + cantidad
                        let totgrl_r = totgrl_r + cantidad
              when
                   r05r.rp2_ident = "U"
                        let ident_u  = ident_u + cantidad
                        let totid_u  = totid_u + cantidad
                        let totgrl_u = totgrl_u + cantidad
              when
                   r05r.rp2_ident = "A"
                        let ident_a  = ident_a + cantidad
                        let totid_a  = totid_a + cantidad
                        let totgrl_a = totgrl_a + cantidad
        end case
        case
              when
                   r05r.rp2_sexo = "F"
                        let mujeres = mujeres      + cantidad
                        let totmuj = totmuj        + cantidad
                        let totgrlmuj = totgrlmuj + cantidad
              when
                   r05r.rp2_sexo = "M"
                        let hombres = hombres      + cantidad
                        let tothom = tothom        + cantidad
                        let totgrlhom = totgrlhom + cantidad
        end case
after group of r05r.rp2_adscr
        let total_id = ident_r + ident_u + ident_a
        let total = hombres + mujeres
        skip 3 line
        print column 1, "TOTAL POR ADSCRIPCION - - - -->","      MUJERES : ",
                         mujeres,"    HOMBRES : ", hombres,
                         "       IDENTIFICADOR R : ",ident_r,
                         "       IDENTIFICADOR U : ",ident_u,
                         "       IDENTIFICADOR A : ",ident_a,
              column 194, "TOTAL : ",total
```

```
        skip 1 line
        print column 1, "REGISTROS IMPRESOS  " group count(*) using
        "####"
after group of r05y_rp2_tnomi
        let totals_id = totid_f  + totid_u  + totid_a
        let totgrl_id = totgrl_r + totgrl_u + totgrl_a
        let totales   = totmuj   + totnom
        let totalgnrl = totgrlmuj + totgrlnom
        skip 3 line
        print column 1, "TOTAL PCS TIPO DE NOMINA . . . . .>"      MUJERES : ".
                        totmuj,"    HOMBRES : ",totnom,
                        "      IDENTIFICADOR R : ",totid_f,
                        "      IDENTIFICADOR U : ",totid_u,
                        "      IDENTIFICADOR A : ",totid_a,
              column 194,"TOTAL : ",totales
        skip 1 line
        print column 1, "TOTAL DE REGISTROS IMPRESOS : ",group count(*) using
        "######"
on last row
        skip 3 line
        print column 1, "TOTAL GENERAL . . . . .>"      MUJERES : ".
                        totgrlmuj,"    HOMBRES : ", totgrlnom,
                        "      IDENTIFICADOR R : ",totgrl_r,
                        "      IDENTIFICADOR U : ",totgrl_u,
              column 194,"TOTAL : ",totalgnrl
        print column 1, "TOTAL GENERAL DE REGISTROS IMPRESOS : ",count(*) using
        "########"
end report
```

```
-- ****************************************
-- * SISTEMA  : Uniformes y ropa de trabajo *
-- * PROGRAMA : rop06r.4gl                  *
-- * OBJETIVO : Reporte del catalogo de     *
-- *            ropa de trabajo con homolo- *
-- *            gacion de donaciones        *
-- * AUTOR    : Monica Ramirez Hernandez    *
-- * FECHA    : 23 de agosto de 1996        *
 * ****************************************

GLOBALS
        "ropglt.4gl"

FUNCTION rop06r()
define vm_numcma     smallint
define vm_numcmachr char(2)
define vm_nomarch    varchar(20)
define vm_nombre     varchar(100)
define vm_comando    varchar(50)
               MENU "Enviar Reporte a "
                    COMMAND "Impresora" "Envia Reporte a la Impresora"
                            call cres_nomarch("rop06r") returning vm_nomarch
                               call imp_imp(10,10,vm_nomarch) returning vm_comando
                               START REPORT rep_cat TO vm_nomarch
                               CALL rop06ra()
                               call manda_impre(vm_comando,vm_nomarch)
                               call borra_arch(vm_nomarch)
                    COMMAND "Archivo" "Envia Reporte a un Archivo"
                               call pide_nombre(10,10) returning vm_nombre
                               if vm_nombre is null then
                                       next option "Salida"
                               else
                                  START REPORT rep_cat TO vm_nombre
                                  CALL rop06ra()
                                  next option "Reimpresion"
                               end if
                    COMMAND "Pantalla" "Envia Reporte a la Pantalla"
                               call cres_nomarch("rop06r") returning vm_nomarch
                               START REPORT rep_cat TO vm_nomarch
                               CALL rop06ra()
                               call more_archivo(vm_nomarch)
                    COMMAND "Reimpresion" "Envia a impresora el reporte generado en la opcion <Archivo>"
                               call pide_nombre(10,10) returning vm_nombre
                               if vm_nombre is null then
                                       next option "Salida"
                               else
                                  call imp_imp(10,10,vm_nombre) returning vm_comando
                                  call manda_impre(vm_comando,vm_nombre)
                                  call borra_arch(vm_nombre)
                               end if
                    COMMAND "Salida" "Salir al Menu Principal"
                               EXIT MENU
       END MENU
END FUNCTION

FUNCTION rop06ra()

   DEFINE rhomo record like ropa20.*
   DEFINE corte char(100)

   OPEN WINDOW wre AT 2,1
      WITH FORM "frop06r"

   LET int_flag = FALSE
   LET corte = " "

   INPUT BY NAME gr_rop.sdac1,gr_rop.sdac2,gr_rop.cate1,
                 gr_rop.cate2,gr_rop.nomi1,gr_rop.nomi2,
                 gr_rop.sexo1,gr_rop.sexo2,gr_rop.ejerc

      AFTER FIELD sdac1
         IF gr_rop.sdac1 = " " or gr_rop.sdac1 is null THEN
            LET gr_rop.sdac1 = "00000000"
            LET gr_rop.sdac2 = "99999999"
            DISPLAY BY NAME gr_rop.sdac1,gr_rop.sdac2
            NEXT FIELD cate1
         END IF

      AFTER FIELD sdac2
         IF (gr_rop.sdac2 = " " or gr_rop.sdac2 is null) AND
            gr_rop.sdac1 <> " " THEN
            LET gr_rop.sdac2 = gr_rop.sdac1
            DISPLAY BY NAME gr_rop.sdac2
            NEXT FIELD cate1
         END IF

      AFTER FIELD cate1
         IF gr_rop.cate1 = " " or gr_rop.cate1 is null THEN
            LET gr_rop.cate1 = "00000"
            LET gr_rop.cate2 = "99999"
            DISPLAY BY NAME gr_rop.cate1,gr_rop.cate2
            NEXT FIELD nomi1
         END IF

      AFTER FIELD cate2
         IF (gr_rop.cate2 = " " or gr_rop.cate2 is null) AND
            gr_rop.cate1 <> " " THEN
            LET gr_rop.cate2 = gr_rop.cate1
            DISPLAY BY NAME gr_rop.cate2
            NEXT FIELD nomi1
         END IF

      AFTER FIELD nomi1
         IF gr_rop.nomi1 = " " or gr_rop.nomi1 is NULL  THEN
            LET gr_rop.nomi1 = "1"
                LET gr_rop.nomi2 = "9"
                DISPLAY BY NAME gr_rop.nomi1, gr_rop.nomi2
           NEXT FIELD sexo1
         END IF

      AFTER FIELD nomi2
         IF (gr_rop.nomi2 = " " or gr_rop.nomi2 is NULL) AND
               gr_rop.nomi1 <> " " THEN
               LET gr_rop.nomi2 = gr_rop.nomi1
```

```
            DISPLAY BY NAME gr_rop.nom12
            NEXT FIELD sexo1
         END IF

      AFTER FIELD nom12
         IF gr_rop.nom12 < gr_rop.nom11 THEN
            ERROR "El segundo cipo de nomina es menor al primero..."
            NEXT FIELD nom12
         END IF

      AFTER FIELD sexo1
         IF gr_rop.sexo1 = " " or gr_rop.sexo1 is NULL THEN
            ERROR "El valor es invalido,..verifique..."
            NEXT FIELD sexo1
         ELSE
            NEXT FIELD sexo2
         END IF

      AFTER FIELD sexo2
         IF gr_rop.sexo2 = " " or gr_rop.sexo2 is NULL THEN
            ERROR "El valor es invalido.. verifique..."
            NEXT FIELD sexo2
         ELSE
            NEXT FIELD ejerc
         END IF

      AFTER FIELD ejerc
         IF gr_rop.ejerc  = "0" or gr_rop.ejerc is null THEN
            ERROR "El campo no puede Quedar en blanco. verifique..."
            NEXT FIELD ejerc
         END IF
   END INPUT

   IF int_flag = TRUE THEN
      LET int_flag = FALSE
      CLEAR FORM
      MESSAGE "Generacion de Reporte Cancelado"
      SLEEP 2
      RETURN
   END IF

   error "Favor de esperar ...  Gracias"

   LET int_flag = FALSE

   DECLARE listcat CURSOR FOR
      SELECT *
      FROM   ropa20
      WHERE  rp2_adscr between gr_rop.adscr1 and gr_rop.adscr2 AND
             rp2_categ between gr_rop.categ1 and gr_rop.categ2 AND
             rp2_tnom1 between gr_rop.nom11 and gr_rop.nom12 AND
             rp2_sexo  between gr_rop.sexo1 and gr_rop.sexo2

      FOREACH listcat INTO rhomo.*
         let corte = rhomo.rp2_sexo,rhomo.rp2_cverh1,rhomo.rp2_cant11,
                     rhomo.rp2_cverh2,rhomo.rp2_cant12,rhomo.rp2_cverh3,
                     rhomo.rp2_cant13,rhomo.rp2_cverh4,rhomo.rp2_cant14,
                     rhomo.rp2_cverh5,rhomo.rp2_cant15,rhomo.rp2_cverh6,
                     rhomo.rp2_cant16,rhomo.rp2_cverh7,rhomo.rp2_cant17,
                     rhomo.rp2_cverh8,rhomo.rp2_cant18,rhomo.rp2_cverh9,
                     rhomo.rp2_cant19,rhomo.rp2_cverh10,rhomo.rp2_cant110,
                     rhomo.rp2_cverh11,rhomo.rp2_cant111,rhomo.rp2_cverh12,
                     rhomo.rp2_cant112,rhomo.rp2_cverh13,rhomo.rp2_cant113,
                     rhomo.rp2_cverh14,rhomo.rp2_cant114,rhomo.rp2_cverh15,
                     rhomo.rp2_cant115

      OUTPUT TO REPORT rep_cat(rhomo.*,corte)
      END FOREACH
      FREE listcat
      FINISH REPORT rep_cat
END FUNCTION

REPORT rep_cat(rhomo20)

DEFINE rhomo20 record
rp2_tnom1 char(1),rp2_adscr char(6),rp2_categ char(5),rp2_sexo char(2),
rp2_cont smallint,rp2_ident char(11),rp2_cverh1 char(4),rp2_cant11 char(2),
rp2_cverh2 char(4),rp2_cant12 char(2),rp2_cverh3 char(4),rp2_cant13 char(2),
rp2_cverh4 char(4),rp2_cant14 char(2),rp2_cverh5 char(4),rp2_cant15 char(2),
rp2_cverh6 char(4),rp2_cant16 char(2),rp2_cverh7 char(4),rp2_cant17 char(2),
rp2_cverh8 char(4),rp2_cant18 char(2),rp2_cverh9 char(4),rp2_cant19 char(2),
rp2_cverh10 char(4),rp2_cant110 char(2),rp2_cverh11 char(4),rp2_cant111 char(2),
rp2_cverh12 char(4),rp2_cant112 char(2),rp2_cverh13 char(4),rp2_cant113 char(2),
rp2_cverh14 char(4),rp2_cant114 char(2),rp2_cverh15 char(4),rp2_cant115 char(2),
rcorte char(100)
END RECORD

   DEFINE leysdsc LIKE emadscr1.descr1
   DEFINE leycate LIKE emcatego.descr1
   DEFINE descrop LIKE ropa10.rp1_desc
   DEFINE cvesol LIKE ropa10.rp1_cvesol

{
   OUTPUT
      LEFT MARGIN 0
      TOP MARGIN 1
      BOTTOM MARGIN 0
}

--   ORDER BY rhomo20.rp2_adscr,rhomo20.rp2_tnom1,rhomo20.rp2_categ,rhomo20.rcorte
   ORDER BY rhomo20.rp2_adscr,rhomo20.rcorte
                  -- rhomo20.rp2_sexo,

   FORMAT
      PAGE HEADER

         SELECT descr1 INTO leysdsc
         FROM   emadscr1
         WHERE  emadscr1.depetc = rhomo20.rp2_adscr

            PRINT COLUMN 3, "ropo41r",
              COLUMN 17, "SISTEMA DE TRANSPORTE COLECTIVO (METRO)",
              COLUMN 60, "FECHA : ", TODAY USING "dd/mm/yyyy"
         SKIP 1 LINE
```

```
            PRINT COLUMN 01, "EJERCICIO : ",year(today),
--          PRINT COLUMN 01, "EJERCICIO : 1994",
              COLUMN 23, "CATALOGO DE ROPA DE TRABAJO",
              COLUMN 60, "HOJA : ", PAGENO USING "###"
         PRINT COLUMN 22, "CON HOMOLOGACION DE DOTACIONES"
         SKIP 1 LINE
         PRINT COLUMN 03, "ADSCRIPCION : ", rhomo20.rp2_adscr,5 SPACES, leyadsc

         PRINT COLUMN 01,"¬-------------------¬",
               COLUMN 21,"¬-------------------¬",
               COLUMN 41,"¬-------------------¬",
               COLUMN 61,"¬-------------------¬"

            PRINT COLUMN 05, "C A T E G O R I A",
               COLUMN 30, "S E X O"
            PRINT COLUMN 15, "P R E N D A",
               COLUMN 42, "CVE. R.M.",
               COLUMN 55, "CVE. ALM.",
               COLUMN 67, "CANT."

         PRINT COLUMN 01,"¬-------------------¬",
               COLUMN 21,"¬-------------------¬",
               COLUMN 41,"¬-------------------¬",
               COLUMN 61,"¬-------------------¬"

BEFORE GROUP OF rhomo20.rp2_adscr
   SKIP TO TOP OF PAGE

--BEFORE GROUP OF rhomo20.rcorte
  ON EVERY ROW
   SELECT descri INTO leycate
   FROM   emcatego
   WHERE emcatego.catetc = rhomo20.rp2_categ
   WEED 15 LINES
   PRINT COLUMN 01, rhomo20.rp2_categ,
         COLUMN 08, leycate,
         COLUMN }}, rhomo20.rp2_sexo
   -- SKIP 1 LINE

--ON EVERY ROW
AFTER GROUP OF rhomo20.rcorte
   SKIP 2 LINES
      IF (rhomo20.rp2_cverh1 is not null AND rhomo20.rp2_cverh1 > 0) THEN
         SELECT rp1_desc,rp1_cveal into descrop,cveal
         FROM ropa10
         WHERE rhomo20.rp2_cverh1 = rp1_cverh
         PRINT COLUMN 10,descrop,
               COLUMN 45, rhomo20.rp2_cverh1,
               COLUMN 55, cveal,
                        COLUMN 66, rhomo20.rp2_cant11
      END IF

      IF (rhomo20.rp2_cverh2 is not null AND rhomo20.rp2_cverh2 > 0) THEN
         SELECT rp1_desc,rp1_cveal into descrop,cveal
         FROM ropa10
         WHERE rhomo20.rp2_cverh2 = rp1_cverh
         PRINT COLUMN 10,descrop,
               COLUMN 45, rhomo20.rp2_cverh2,
               COLUMN 55, cveal,
                        COLUMN 68, rhomo20.rp2_cant12
      END IF

      IF (rhomo20.rp2_cverh3 is not null AND rhomo20.rp2_cverh3 > 0) THEN
         SELECT rp1_desc,rp1_cveal into descrop,cveal
         FROM ropa10
         WHERE rhomo20.rp2_cverh3 = rp1_cverh
         PRINT COLUMN 10,descrop,
               COLUMN 45, rhomo20.rp2_cverh3,
               COLUMN 55, cveal,
                        COLUMN 69, rhomo20.rp2_cant13
      END IF

      IF (rhomo20.rp2_cverh4 is not null AND rhomo20.rp2_cverh4 > 0) THEN
         SELECT rp1_desc,rp1_cveal into descrop,cveal
         FROM ropa10
         WHERE rhomo20.rp2_cverh4 = rp1_cverh
         PRINT COLUMN 10,descrop,
               COLUMN 45, rhomo20.rp2_cverh4,
               COLUMN 55, cveal,
                        COLUMN 64, rhomo20.rp2_cant14
      END IF

      IF (rhomo20.rp2_cverh5 is not null AND rhomo20.rp2_cverh5 > 0) THEN
         SELECT rp1_desc,rp1_cveal into descrop,cveal
         FROM ropa10
         WHERE rhomo20.rp2_cverh5 = rp1_cverh
         PRINT COLUMN 10,descrop,
               COLUMN 45, rhomo20.rp2_cverh5,
               COLUMN 55, cveal,
                        COLUMN 68, rhomo20.rp2_cant15
      END IF

      IF (rhomo20.rp2_cverh6 is not null AND rhomo20.rp2_cverh6 > 0) THEN
         SELECT rp1_desc,rp1_cveal into descrop,cveal
         FROM ropa10
         WHERE rhomo20.rp2_cverh6 = rp1_cverh
         PRINT COLUMN 10,descrop,
               COLUMN 45, rhomo20.rp2_cverh6,
               COLUMN 55, cveal,
                        COLUMN 68, rhomo20.rp2_cant16
      END IF

      IF (rhomo20.rp2_cverh7 is not null AND rhomo20.rp2_cverh7 > 0) THEN
         SELECT rp1_desc,rp1_cveal into descrop,cveal
         FROM ropa10
         WHERE rhomo20.rp2_cverh7 = rp1_cverh
         PRINT COLUMN 10,descrop,
               COLUMN 45, rhomo20.rp2_cverh7,
               COLUMN 55, cveal,
                        COLUMN 68, rhomo20.rp2_cant17
      END IF

      IF (rhomo20.rp2_cverh8 is not null AND rhomo20.rp2_cverh8 > 0) THEN
         SELECT rp1_desc,rp1_cveal into descrop,cveal
```

```
            FROM ropa10
            WHERE rhomo20.rp2_cverh8 = rp1_cverh
            PRINT COLUMN 10,descrop
                  COLUMN 45, rhomo20.rp2_cverh8,
                  COLUMN 55, cveal,
                             COLUMN 68, rhomo20.rp2_cant18
         END IF

         IF (rhomo20.rp2_cverh9 is not null AND rhomo20.rp2_cverh9 > 0) THEN
            SELECT rp1_desc,rp1_cveal into descrop,cveal
            FROM ropa10
            WHERE rhomo20.rp2_cverh9 = rp1_cverh
            PRINT COLUMN 10,descrop,
                  COLUMN 45, rhomo20.rp2_cverh9,
                  COLUMN 55, cveal,
                             COLUMN 68, rhomo20.rp2_cant19
         END IF

         IF (rhomo20.rp2_cverh10 is not null AND rhomo20.rp2_cverh10 > 0) THEN
            SELECT rp1_desc,rp1_cveal into descrop,cveal
            FROM ropa10
            WHERE rhomo20.rp2_cverh10 = rp1_cverh
            PRINT COLUMN 10,descrop,
                  COLUMN 45, rhomo20.rp2_cverh10,
                  COLUMN 55, cveal,
                             COLUMN 68, rhomo20.rp2_cant110
         END IF

         IF (rhomo20.rp2_cverh11 is not null AND rhomo20.rp2_cverh11 > 0) THEN
            SELECT rp1_desc,rp1_cveal into descrop,cveal
            FROM ropa10
            WHERE rhomo20.rp2_cverh11 = rp1_cverh
            PRINT COLUMN 10,descrop,
                  COLUMN 45, rhomo20.rp2_cverh11,
                  COLUMN 55, cveal,
                             COLUMN 68, rhomo20 rp2_cant111
         END IF

         IF (rhomo20.rp2_cverh12 is not null AND rhomo20.rp2_cverh12 > 0) THEN
            SELECT rp1_desc,rp1_cveal into descrop,cveal
            FROM ropa10
            WHERE rhomo20.rp2_cverh12 = rp1_cverh
            PRINT COLUMN 10,descrop,
                  COLUMN 45, rhomo20.rp2_cverh12,
                  COLUMN 55, cveal,
                             COLUMN 68, rhomo20.rp2_cant112
         END IF

         IF (rhomo20.rp2_cverh13 is not null AND rhomo20.rp2_cverh13 > 0) THEN
            SELECT rp1_desc,rp1_cveal into descrop,cveal
            FROM ropa10
            WHERE rhomo30 rp2_cverh13 = rp1_cverh
            PRINT COLUMN 10,descrop,
                  COLUMN 45, rhomo20.rp2_cverh13,
                  COLUMN 55, cveal
                             COLUMN 68, rhomo20.rp2_cant113
         END IF

         IF (rhomo20.rp2_cverh14 is not null AND rhomo20.rp2_cverh14 > 0) THEN
            SELECT rp1_desc,rp1_cveal into descrop,cveal
            FROM ropa10
            WHERE rhomo20.rp2_cverh14 = rp1_cverh
            PRINT COLUMN 10,descrop,
                  COLUMN 45, rhomo20.rp2_cverh14,
                  COLUMN 55, cveal,
                             COLUMN 68, rhomo20.rp2_cant114
         END IF

         IF (rhomo20.rp2_cverh15 is not null AND rhomo20.rp2_cverh15 > 0) THEN
            SELECT rp1_desc,rp1_cveal into descrop,cveal
            FROM ropa10
            WHERE rhomo20.rp2_cverh15 = rp1_cverh
            PRINT COLUMN 10,descrop,
                  COLUMN 45, rhomo20.rp2_cverh15,
                  COLUMN 55, cveal,
                             COLUMN 68, rhomo20 rp2_cant115
         END IF
         SKIP 2 LINES

      ON LAST ROW
         SKIP 4 LINES
         PRINT COLUMN 05, "TOTAL GENERAL ---> ",
               COLUMN 10, COUNT/*| USING "##,###"
   END REPORT
```

GLOSARIO

Portatilidad :

Facilidad con la que un producto de programación puede ser transferido de un sistema de cómputo a otro o de un ambiente a otro.

Confiabilidad :

Capacidad de un programa de realizar una función requerida bajo ciertas condiciones durante un periodo determinado.

Eficiencia :

Grado con el que un producto de programación efectúa sus funciones, mediante un mínimo de recursos computacionales.

Exactitud :

Especificación cualitativa de ausencia de error. Medida cuantitativa de la magnitud del error.

Error :

Discrepancia entre una condición o valor calculado y la condición real, especificada o valor correcto teórico.

Solidez :

Grado con el que un producto de programación puede continuar operando correctamente, a pesar de la introducción de datos inválidos.

Corrección :

Grado en el que un producto de programación esta libre de defectos de diseño y de codificación, esto es, libre de faltas. Grado en que un producto de programación cumple los requisitos especificados. Grado en que un producto de programación cumple con las expectativas del usuario.