UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ARAGÓN

CONSTRUCCIÓN DE UN MODELO DE RENTABILIDAD FINANCIERO BASADO EN UN DATA WAREHOUSE

TRABAJO BAJO LA MODALIDAD
DE INFORME DE EJERCICIO PROFESIONAL

PARA OBTENER EL TITULO DE

INGENIERO EN COMPUTACIÓN

P R E S E N T A :

MARICELA SOTO DE LUNA

ASESOR: ING. ERNESTO PEÑALOZA ROMERO

MAYO 2007

Agradecimientos

A Dios por obsequiarme Salud e Inteligencia.

A mis Padres Graciela y Lorenzo por su amor incondicional, saben que todos mis éxitos son suyos porque uds. me enseñaron a tener sueños y alcanzarlos.

A mis hermanos Ale, Ricardo, Lety, Pepi por su cariño y apoyo que me han brindado en todo momento siempre alentándome a seguir adelante.

A mi hijo Gonzalo por entenderme, quererme y sobre todo por ser mi sol que me da la suficiente fuerza y potencia para estar siempre motivada.

A toda mi familia que aunque a distancia siempre están al tanto de mi y me demuestran su cariño cuando lo he necesitado.

A mis amigos (sobre todo a Rosario, Celia, Uriel y David) que me regalan su amista y cariño y siempre han deseado lo mejor para mi.

A todos mis colegas de trabajo y a mis jefes por su amistad, confianza, sus enseñanzas han sido valiosas para mi son realmente personas talentosas, juntos hemos trabajo en construir procesos éxitos que me han ayuda a consolidar conocimientos y crecer profesionalmente.

A mis profesores espero ser siempre una buena profesionista que represente orgullosamente a la UNAM.

A mi esposo Gonzalo de León A por amarme y creer en mi, su compañía, amor y apoyo son mi principal pilar en mi vida.

ÍNDICE

CONTENIDO ..... PÁGINA

1.INTRODUCCIÓN 1

2. MARCO TEÓRICO 3

3.OBJETIVO DE NEGOCIO: Modelo de Rentabilidad 18

4. ESTRATEGIA DE SOLUCIÓN 23

5. DISEÑO 29

6. CONSTRUCCIÓN 39

7. USO Y MONITOREO 48

8.EVOLUCIÓN 52

9. CONCLUSIÓN 60

APÉNDICE 1: Conceptos Base de Datos Relacional 65

APÉNDICE 2: Data warehouse y esquemas de fragmentación. . 69

APÉNDICE 3: Nociones Básicas de SQL embebido en C 72

APÉNDICE 4: El problema de transferencia, solución a un problema crítico. 77

1.INTRODUCCIÓN

En esta era actual de la información, uno de los mayores problemas a los que las empresas se enfrentan es el tener mucha información y no saber que hacer con ella. Las empresas hoy en día requieren que la información relevante se encuentre disponible en tiempo y forma para de esta manera, dirigir el negocio de acuerdo con las necesidades cambiantes del mercado; requieren adquirir la habilidad de adaptarse a los cambios en el entorno, para adquirir ventajas competitivas.

En 1996, Banco[1] Bital enfrentaba problemas para integrar datos que provenían de diversas fuentes, los gerentes de producto invertían mucho tiempo para recabar información desde los sistemas origen, por lo que dedicaban muy poco tiempo al análisis y diseño de productos, definición de campañas y por consiguiente a la estrategia de ventas. Se requería distribuir información a todos sus colaboradores para facilitar la toma de decisiones a la Dirección.

Las ideas aquí expuestas son resultado de mi experiencia laboral de 5 años en un caso real dentro del Banco para dar solución al problema de integrar datos para la buena toma de decisiones, donde estaba a cargo de una gerencia y a mi cargo 8 personas; se describirán las principales responsabilidades de la gerencia que se resumen enseguida:

1. Administración de la información (datos) : Función primordial del área decidir con exactitud cuál es la información que debe mantenerse en las Bases de Datos, es decir, identificar las entidades de interés común de los usuarios, realizando un estudio previo de necesidades de información de nuestros cliente, para elaborar un Diseño lógico o conceptual de la Base de Datos, ya sea para actualizar, borrar ó ingresar información, y cuándo aplicaba diseñar el modelo físico con asesoramiento de un DBA.
2. Establecimiento de políticas para mantener la integridad de la información: Estableciendo las políticas para mantener, manejar los datos y mantener la seguridad de la información.

En cada etapa del documento se reflejará el rol que desempeñe que fue de Arquitecto de datos, el principal objetivo de este rol es reforzar la arquitectura del Data Warehouse, en cada una de las iteraciones que ha determinado el negocio. Dentro de las responsabilidades de este rol están las siguientes:

- Trabajar estrechamente con el Analista de Negocio para entender y documentar las necesidades de los usuarios, para identificar y analizar los datos de apoyo a las decisiones, esto incluye Actuar como el enlace entre el Equipo de trabajo Data Warehouse IT y los usuarios finales del DW.
- Entender los requisitos de información por lo que se refiere a las unidades de negocio específicas. Traduciendo y descomponiendo esos requisitos para que cada uno de los participantes tanto de Negocio y técnico lo entienda cada quien en su contexto. Asegurar que las necesidades de negocio se incorporen a las soluciones técnicas definidas.

[1] **Banco** Un banco es una entidad que ofrece un lugar para guardar su dinero y emplea el dinero para hacer más dinero. Los bancos ofrecen diferentes servicios a cambio del dinero del ahorrador.

- Diseñó del proceso de Adquisición de Datos. Identificar fuentes apropiadas de datos para alimentar el Data Warehouse, Identificar oportunamente los obstáculos involucrados para obtener los datos necesitados por el negocio
- Modelado de datos lógico para un Data Warehouse (con el conocimiento de las diferencias entre un ambiente Data Warehouse y un ambiente operacional tradicional). Esta responsabilidad incluye la identificación de datos más valiosos a la corporación, la integración de estos datos, y el mapeo del dato para ponerlo en correlación al modelo de los datos.
- Participar en la comprobación del Data Warehouse. Validar que la solución del Data Warehouse satisface los requisitos de los usuarios finales, es decir, asegurar que el extracto de datos del sistema fuente y el movimiento al Data Warehouse se realicen "correctamente", y se realicen las transformaciones u otras reglas de negocio aplicadas a los datos "correctamente." Correctamente se define como "de acuerdo con los requisitos de usuarios finales identificados".
- Documentar procesos y los datos del Data Warehouse.
- Estimar el tamaño de la aplicación del Data Warehouse , Estimación de costos para la soluciones técnicas y evaluación y selección de hardware y software

Adicionalmente este informe proveerá al lector la ilustración de un caso real con los pasos que se siguieron para lograr el desarrollo de un modelo de datos de rentabilidad soportado por una plataforma de data warehouse. Esto no implica generar un modelo compuesto por herramientas complejas y costosas; la propuesta es alinear los objetivos de la empresa, en materia de optimización de sus recursos, generando un modelo de toma de decisiones flexible, acorde al tamaño de la organización, una herramienta que le permita tomar decisiones para la prevención de problemas y detección de áreas de oportunidad. A lo largo de este trabajo el lector podrá reforzar todos los términos de Inteligencia de Negocio [Business intelligence][2].

"La inteligencia de negocio ayuda a encontrar lo que en realidad funciona y lo que no" Bill Gates. Director de Microsoft.

Como antecedente a la implementación del modelo de datos de rentabilidad, el Banco desarrollo en Noviembre de 1995 algunos procesos que alimentaban un gran repositorio de datos, intentando dar respuesta a la creciente necesidad de información oportuna y confiable, cuyo objetivo primordial fue robustecer los temas de direccionamiento estratégico de toma de decisiones de Bital; sin embargo No se obtuvieron los resultados deseados debido a la poca experiencia en sistemas de información para la toma de decisiones y la complejidad del ambiente tecnológico requerido para una solución de esta naturaleza. Con base a esta experiencia se integro un grupo de trabajo dedicado al proyecto denominado Data Warehouse. La primera etapa de este proyecto se enfocó en cubrir necesidades de segmentación de mercado y de evaluación de rentabilidad de los clientes, misma que es el alcance del presente documento.

[2] **Inteligencia de negocios (BI)** se puede definir como el proceso para analizar datos acumulados de la empresa y adquirir cierto conocimiento de ellos.

2. MARCO TEÓRICO

En los siguientes párrafos se describe el marco teórico, para facilitar al lector la introducción de algunos términos y la comprensión de los antecedentes generales que dieron origen a los sistemas de información llamados Data Warehouse[3].

Durante los últimos años, los sistemas de información se han convertido en uno de los principales ámbitos de estudio en el área de organización de empresas. El entorno donde las compañías desarrollan sus actividades se vuelve cada vez más complejo. La creciente globalización, el proceso de internacionalización de la empresa, la rapidez en el desarrollo de las tecnologías de información, el aumento de la certidumbre en el entorno y la reducción de los ciclos de vida de los productos hacen de la información un elemento clave para la gestión así como para la supervivencia y crecimiento de la organización; la información aparece como un insumo fundamental generador de valor en las empresas.

En la actualidad, la diferencia entre las empresas es cada vez menor en cuanto a recursos se refiere, la diferencia estriba en el valor que ésta le proporciona al producto o servicio que ofrece. Para lograr lo anterior, se pone énfasis en otros factores competitivos como calidad, tiempo de respuesta, diseño a la medida, innovación, etc. Uno de los principales objetivos de los sistemas de información es precisamente la mejora de resultados obtenidos por la empresa en estos factores clave de competitividad.

Para comprender en toda su extensión las implicaciones de la implantación de un sistema de información, es necesario hacer primero algunas definiciones y precisiones básicas.

Sistema de información: "Conjunto formal de procesos que, operando sobre una colección de datos estructurada de acuerdo a las necesidades de la empresa, recopila, elabora y distribuyen selectivamente la información necesaria para la operación de dicha empresa y para las actividades de dirección y control correspondientes, apoyando en parte los procesos de toma de decisiones necesarios para desempeñar funciones de negocio de la empresa de acuerdo con su estrategia".

Todo sistema de información utiliza como materia prima los datos, los cuales almacena, procesa y transforma para obtener como resultado final información, la cual será suministrada a los diferentes usuarios del sistema; existe además un proceso de retroalimentación en el cual se ha de valorar si la información obtenida se adecua a lo esperado.

[3] **Data Warehouse** o Bodega de Datos es un proceso, no un producto. Es una técnica para consolidar y administrar datos de variadas fuentes con el propósito de responder preguntas de negocios y tomar decisiones

Los sistemas de información, a diferencia de las Tecnologías de Información (TI), son un concepto más amplio, pues establecen cuáles son las necesidades de información de las empresas, cómo lo van a solucionar y qué medios (tecnologías de información) van a emplear; un sistema de información abarca más que el aspecto tecnológico, ya que no sólo toma en cuenta las herramientas, sino también el modo de organizar dichas herramientas y de obtener la información necesaria para el correcto funcionamiento de la empresa.

Una característica importante que debe poseer el quipo encargado de elaborar los sistemas de información es el tener conocimientos tanto de las TI disponibles y que pueden utilizarse en la empresa, además del modo de organizarlas. Para ello en primer lugar tendrán que conocer la estrategia y el tipo de organización para posteriormente establecer las necesidades de información y adquirir las herramientas necesarias para el desarrollo del sistema de información.

Todo sistema de información tiene objetivos principales, los cuales se resumen a continuación:

- ✓ Apoyar los objetivos y estrategias de la empresa: el sistema de información ha de suministrar a la organización toda la información necesaria para su correcto funcionamiento.
- ✓ Dar información para la evolución de la empresa: conforme la empresa va creciendo y desarrollándose, surgen nuevas necesidades de información que deberán ser satisfechas por el sistema de información evolucionando este último adecuándose a las nuevas circunstancias del entorno.
- ✓ Interactuar con las diferentes áreas de la organización, permitiendo que estos empleen el sistema de información para satisfacer sus necesidades de un modo rápido y eficaz. La interactividad y flexibilidad de los sistemas de información constituyen un punto clave en el éxito o fracaso.

Evolución de los sistemas de información (de acuerdo a lo que W. H. Inmon menciona en su libro "Building the Data Warehouse"**)**

El procesamiento de información es una práctica muy reciente que tiene sus inicios en los 60´s. A inicios de los años 60´s el mundo de la computación consistía de aplicaciones creadas individualmente (autónomas unas de otras) que corrían en base de archivos maestros, las aplicaciones se caracterizaban por la generación de reportes y programas que usualmente eran elaborados en *cobol*[4]; en esta época las tarjetas perforadas eran comunes. Los archivos maestros se almacenaban en cintas magnéticas las cuales tenían los beneficios de almacenar grandes volúmenes de datos a un bajo precio, pero con la gran desventaja que debían ser accesadas secuencialmente lo que implica que la lectura de la cinta siempre era del 100%, no importando que solo se requiriera un 5% o menos. Además, el realizar una lectura completa de la cinta podía tomar de 20 a 30 minutos dependiendo del volumen de datos contenidos en la cinta.

A mediados de los 60´s el crecimiento de los archivos y cintas magnéticas aumentó de manera considerable. En muy poco tiempo había archivos por todos lados. Este crecimiento generó una enorme cantidad de datos redundantes.

[4] Cobol : Leguaje de programación de tercera generación. El lenguaje COBOL, acrónimo de "COmmon Business Oriented Language" nació del deseo de crear un lenguaje de programación "universal", que pudiera ser usado en cualquier ordenador (en los años 1960 existían numerosos modelos de ordenadores incompatibles entre sí) y que estuviera orientado principalmente a los negocios, es decir, a la llamada informática de gestión. Fuente: http://es.wikipedia.org

La llegada del almacenamiento de acceso directo ó almacenamiento en disco (DASD Direct Access Storage Device); en 1970, fue totalmente diferente del almacenamiento en cinta magnética ya que los datos podían ser leídos directamente. Ya no había necesidad de ir de registro en registro (1, 2,3,..., n), sino que iba directo al registro buscado (n+1). Más adelante, el tiempo requerido para ir al registro (n+1), fue significativamente menor que el tiempo requerido para buscar en una cinta, ya que en un esquema de almacenamiento de acceso directo, el tiempo bajó a una medida en milisegundos.

Con la llegada del DASD, vino un nuevo software conocido como gestor de base de datos (DBMS Data Base Management System[5]). La propuesta de los DBMS fue hacer más sencilla la programación para el almacenamiento y acceso de datos en los DASD. Además los DBMS realizaban tareas como almacenar datos sobre DASD, indexar datos, etc. Y con los DBMS viene el concepto de una "base de datos". En la búsqueda de disminuir la creación de archivos en cintas y de disminuir la redundancia de datos, el término base de datos es definido como: una simple fuente de datos para todos los procesos.

A mediados de los setentas, surgen los sistemas transaccionales o de procesamiento en línea (OLTP Online Transaction Processing[6]), estos sistemas ya contaban con un banco de datos, un software apropiado y amigable, además que facilitó al usuario técnico tener acceso más rápido, con esto la computadora podía ser acostumbrada a manejar cualquier tipo de sistemas como los sistemas de reservaciones, los cajeros de bancos, sistemas del mando industrial, etc. Si el mundo de la computación hubiera permanecido en un estado de archivos de cinta magnético, la mayoría de los sistemas que hoy concebimos no habrían surgido.

Por los años ochenta, aparecen nuevas tecnologías como son las PC´s y las herramientas 4GL. El usuario final empezó a asumir un papel importante en el procesamiento de datos, ya que podía controlar los datos directamente, fuera del dominio del procesador de los datos clásico. Con las PC´s y la tecnología de 4GL vino la noción de que los datos de una empresa pueden servir para más que sólo procesamiento de transacciones en los sistemas. Es entonces cuando nace el concepto *MIS* (*Management Information Systems*), sistemas de información que emplean los datos para manejar la información para la toma de decisiones.

En los noventas un programa llamado "proceso de extracción" empezaba a aparecer. El programa de extracción es el más simple de todos los programas. Estos programas buscan a través de un archivo o base de datos usando algunos criterios de selección y, después de encontrar el dato calificado, transporta los datos a otro archivo o base de datos (Figura 2.1).

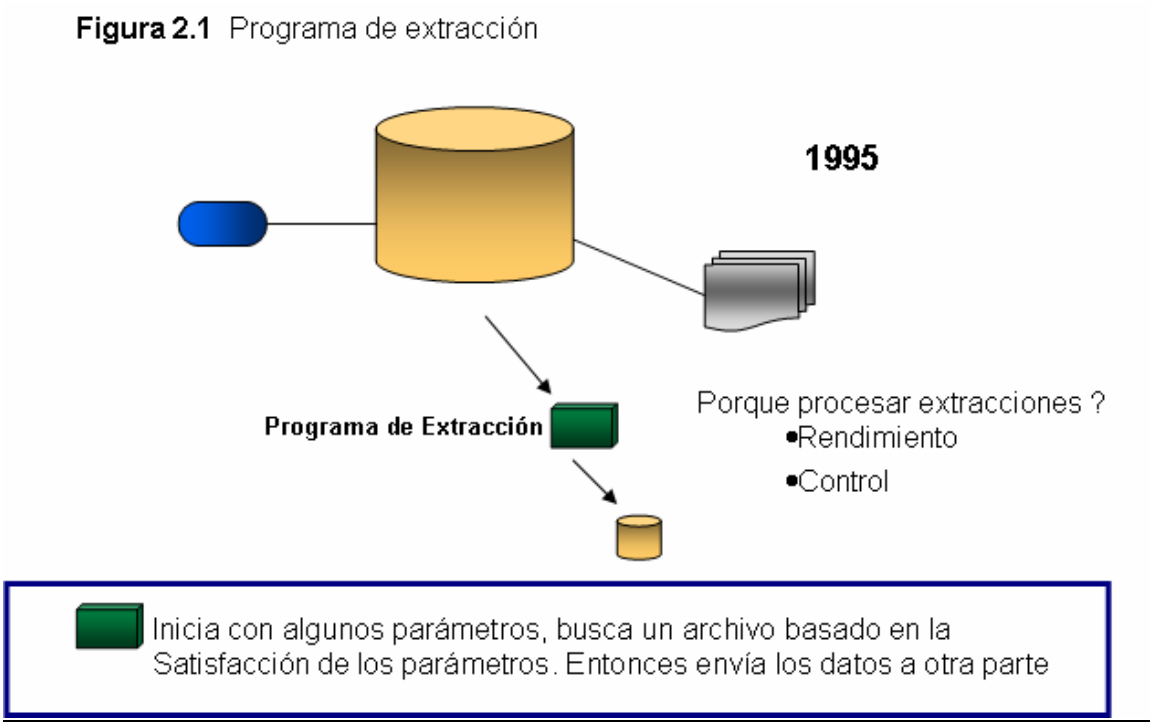
Pronto los programas de extracción empezaron a ser muy populares. Esto permitió un progreso en los ambientes de información. Hay dos razones para la popularidad de estos ambientes:

[5]**DBMS** En español SGBD Sistema de gestión de base de datos, el objetivo de un SGBD es proporcionar un entorno que sea tanto práctico como eficiente de usar en la recuperación y el almacenamiento de la información de la base de datos

[6]**OLTP** (On-Line Transaction Processing) Procesamiento Transaccional en Línea, son sistemas diseñados para conocer las necesidades operacionales del día a día del negocio y el rendimiento de la base de datos esta afinado para este tipo de necesidades operaciones . Consecuentemente la base de datos puede recuperar un pequeño número de registros rápidamente, ya que los sistemas OLTP procesan miles o millones de transacciones por día

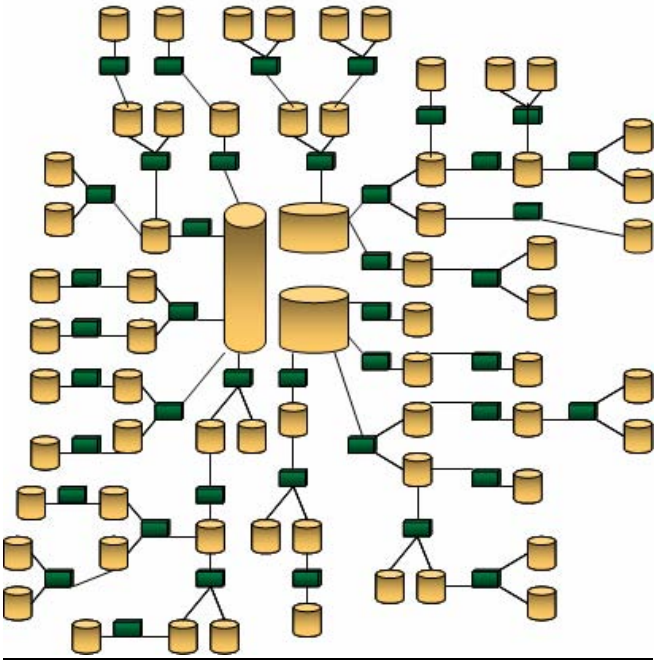
1. Porque los programas de extracción mueven los datos fuera del ambiente de los procesos en línea, es decir del ambiente transaccional.
2. Cuando los datos son movidos fuera del dominio del proceso transaccional con un programa de extracción, hay una transferencia en el control de los datos. El usuario final es el propietario una vez que él ó ella toman el control de los datos.

Figura 2.1 Programa de extracción



Por estas razones, los procesos de extracción pronto se encontraron por todos lados. En los años 90´s había muchos programas de extracción, como se representa en la Figura 2.2

Figura 2.2 “La telaraña”



La Figura 2.2 muestra una “telaraña” de procesos de extracción que empiezan a formase. Primero había extracciones, después había extracciones de las extracciones, después extracciones de las extracciones y así sucesivamente hasta ahora.

Este modelo de proceso de extracción a través de la organización empezó a ser común. Tomando colectivamente, los programas de extracciones se forma una telaraña.

Las principales dificultades asociadas a una telaraña son:

- *A) Falta de credibilidad de los datos*
- *B) Problemas con la Productividad*
- *C) Dificultades para transformar datos en información.*

A) Falta de credibilidad de los datos.

Al no tener una sola fuente de datos es muy factible que diferentes departamentos de una empresa al tomar información de los sistemas operacionales utilicen criterios diferentes para generar reportes similares. El efecto que esto causa en la dirección de la empresa es desconfianza en la información y el riesgo de tomar decisiones erróneas al no tener precisión en los datos.

En adición al problema de no saber en qué reporte confiar, intentar conciliar cifras es una tarea muy difícil de realizar. A menos que se haya realizado un proceso de documentación muy cuidadoso, conciliar estas diferencias es prácticamente imposible.

Cuando la dirección recibe dos reportes con información distinta acerca de las mismas variables, es muy difícil decidir en qué reporte confiar por lo que comúnmente utiliza la toma de decisiones basada en factores subjetivos.

La falta de credibilidad de los datos se da por las siguientes razones:

- Diferentes tiempos para tomar los datos
- Criterios diferentes de extracción
- Diferentes niveles de extracción
- Problemas con datos externos
- Fuentes diferentes de datos

B) Problemas con la productividad

La productividad (o falta de ella), cuando hay una necesidad de buscar datos a través de la organización para consolidarlos en un reporte, y el tiempo para obtenerlos es abismal.

La dirección decide obtener un reporte corporativo. El diseñador asignado a la tarea decide que hay tres cosas que deben hacerse para obtener el reporte corporativo:

- Localizar y analizar los datos para el reporte.
- Obtener y recopilar los datos para el reporte.
- Obtener recursos (analistas programadores para cumplir con la tarea).

Para poder localizar los datos, deben analizarse muchos archivos y layouts de datos, además hay factores que complican la tarea: archivos con elementos con el mismo nombre pero con cifras muy diferentes; otro caso son bases de datos con nombres distintos y la misma información. A menos que los datos sean analizados una y otra vez, el reporte finalizará mezclando información no comparable creando además confusión.

La siguiente tarea, una vez localizados, es recopilar los datos. Sin embargo, para crear el programa para recopilar los datos desde diferentes fuentes, los siguientes factores se deben tomar en cuenta:

- Una gran cantidad de programas deben ser escritos.
- Elaborar programas para cada una de las tecnologías que la organización tiene.

Si el diseñador ha solicitado solo dos o tres personas al mes para la generación del reporte, no distraerá la atención de la dirección. Pero cuando un diseñador solicita muchos recursos, la dirección debe poner la solicitud del reporte corporativo con todos las demás solicitudes que requieren recursos y debe priorizar los requerimientos de tal forma que el diseñador necesita mucho tiempo para satisfacer a su cliente.

Crear el reporte usando una gran cantidad de recursos no debería ser mal visto si hubiera un precedente exitoso. En otras palabras, si a partir del primer reporte corporativo generado que requirió una gran cantidad de recursos, se pudieran generar todos los reportes posteriores, entonces se justificaría pagar el precio , sin embargo muchas veces este no es el caso.

C) Dificultades para transformar datos en información.

Uno de los problemas más serios es la falta de habilidad o conocimiento para transformar los datos en información.

Considerando el siguiente requerimiento de información, típico en un ambiente bancario :

¿Cuál ha sido la rentabilidad de las cuentas diferenciando este año con cada uno de los últimos 5 años?

Lo primero que descubre el analista, al intentar satisfacer el requerimiento de información es que existirán numerosas aplicaciones en los ambientes operacionales que deberá encontrar y analizar.

Intentar exponer qué datos existen para una cuenta es una tarea complicada. Hay aplicaciones de cuentas de Ahorro, hay aplicaciones de créditos/préstamos, hay aplicaciones de seguros, etc. y tratar de seleccionar información común desde estas aplicaciones es virtualmente imposible. Las aplicaciones no fueron construidas pensando en la integración, además no será fácil para los analistas descifrar cada una de las aplicaciones. La integración no es la única dificultad; un obstáculo mayor será que no hay suficiente historia de datos almacenada en las aplicaciones operacionales de la organización.

El departamento de préstamos cuenta con 2 años de datos que pueden ser útiles. Las aplicaciones de cuentas de ahorros cuenta con 60 días de datos ya que las aplicaciones fueron construidas para atender las necesidades del momento o actuales del proceso (suficiente para atender una transacción del cliente o una aclaración) no fueron diseñadas para mantener datos históricos.

Los sistemas encontrados en las arquitecturas de las aplicaciones antes mencionadas son inadecuados para la tarea de soportar la información requerida debido a la falta de integración y la falta de historia.

UN CAMBIO PARA ACERCARSE A LOS SISTEMAS DE SOPORTE A LA DECISIÓN (*DSS DECISION SUPPORT SYSTEM* [7])

Las arquitecturas OLTP y DSS tienen fundamento en dos tipos de datos:

- Datos Primarios
- Datos Derivados

La siguiente tabla muestra algunas de las más representativas diferencias entre los datos primitivos y los datos derivados:

Datos Primarios / Datos Operaciones	Datos Derivados / Datos DSS
Orientado al resultado de una aplicación	Orientado a un tema(s) específico(s)
A detalle	Resumido ó redefinido de otro modo
Actual, como al momento de acceder	Representan un valor en el tiempo, fabricado de repente
Útil para un grupo de personas dentro de la organización	Útil para la Dirección

[7] **DSS** es un recurso intelectual de individuos con el apoyo de la computadora para proveer de decisiones con calidad. El portal de dssresources.com describe un DSS como una gran área de herramientas y sistemas computarizados que ayudan a la toma de decisiones en procesos definidos. DDS es pues una amplia área de análisis que sirve para que la gente examine datos a fin de tomar decisiones ya seas grandes o pequeñas sobre los negocios de su compañía.

Puede ser actualizado	No se puede actualizar
Corre repetitivamente	Corre Heurísticamente[8]
Para procesarlo es necesario entender un requerimiento con anterioridad	Para procesarlo no es necesario entender requerimientos a priori
Compatible con el ciclo de vida de un desarrollo de sistemas (SDLC System development life cycle)	Completamente diferente al ciclo de vida un desarrollo de sistemas
Rendimiento susceptible	Rendimiento relajado (relaxed)
Acceso a una unidad en el tiempo	Acceso a un grupo de datos en el tiempo
Dirigidos a la transaccionalidad	Dirigidos al análisis
Disponibilidad Alta	Disponibilidad Baja
Una prioridad es el control de actualizaciones	El control de actualizaciones no es un resultado
Administración a nivel entidades	Administración a nivel subgrupos de datos
No existe redundancia	La redundancia es un factor indispensable para su funcionalidad
Estructuras estáticas, contenidos variables	Estructuras flexibles
Cantidades pequeñas de datos utilizadas en los procesos	Grandes cantidades de datos utilizados en los procesos
Soporte de operaciones del día a día	Soporte de necesidades de la Dirección
Alta probabilidad de acceso	Baja o modesta probabilidad de acceso

Tabla 2.1 Comparación de datos primitivos y derivados

Los datos primitivos son detallados, son datos utilizados para correr operaciones del día a día de la Empresa. Los datos derivados son datos que son procesados o calculados de otra manera para satisfacer las necesidades de la Dirección de la Empresa. Los datos primitivos pueden ser actualizados, los datos derivados no. Los datos primitivos son principalmente datos con valores actuales. Los datos derivados son frecuentemente datos históricos. Los datos primitivos son operados por procedimientos que corren de forma repetitiva. Los datos derivados son operados por programas que corren heurísticamente sobre bases no repetitivas. Los datos operacionales son datos primitivos. Los datos DSS son datos derivados. Los datos primitivos soportan las funciones de la oficina ó de un grupo dentro de la organización. Los datos derivados soportan las funciones directivas.

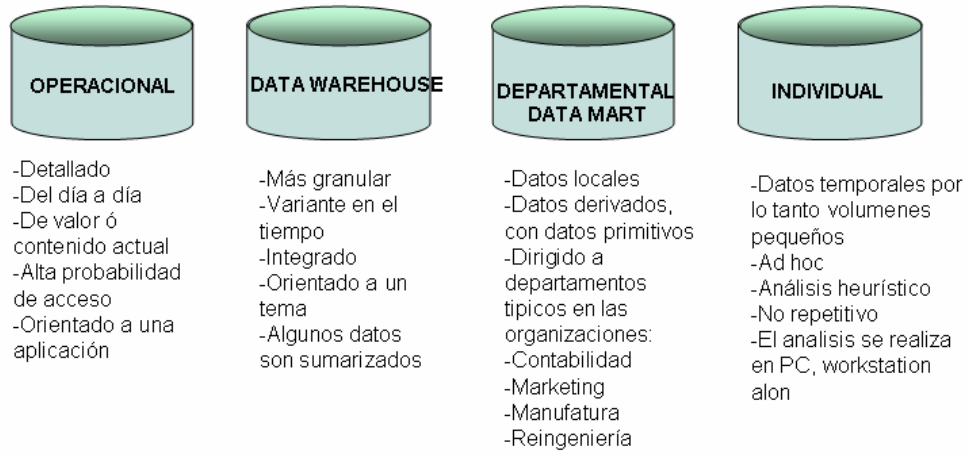
EL AMBIENTE DE ARQUITECTURA PARA ENTENDER COMO CAMBIAR A UNA ARQUITECTURA DE DATA WAREHOUSE*:* Para entender la división entre los datos primitivos y los datos derivados es importante revisar cuatro niveles:

- El operacional
- El data warehouse
- El departamental ó de Data Mart
- El individual

El nivel operacional de datos mantiene sólo datos primitivos que son utilizados por procesos altamente transaccionales. El nivel data warehouse mantiene datos primitivos que no son actualizados, también pueden encontrarse algunos datos derivados en este nivel. El nivel departamental contiene datos derivados. Y el nivel individual de datos es donde se realizan muchos análisis heurísticos. Estas diferencias de muestran en la figura 2.3.

[8] **Heurístico**: Algoritmo que consta de utilizar pruebas, exámenes o aproximaciones para llegar a dar con una solución. De esta forma, sin conocer unos datos base exactos, podemos llegar a un resultado final.

Figura 2.3
NIVELES DE LA ARQUITECTURA



Tener una arquitectura de datos con los niveles mencionados nos da como primera reacción que existe mucha redundancia de datos. Pero este no es el caso si la comparamos con los problemas que se tenían en la arquitectura anterior "la telaraña".

Para entender cómo pasan los datos a los largo de la arquitectura consideremos el ejemplo de la figura 4. En el ambiente operacional existe un solo registro para el cliente, Juan López. El registro en este nivel operacional contiene sólo el valor más actual ó reciente. Para entender el estatus del cliente en este momento, obviamente si la información para Juan López cambia, el registro en el ambiente operacional debe cambiar para reflejar el dato nuevo que será el correcto cuando se requiera saber el estatus actual del cliente.

En el ambiente Data Warehouse se encontrarán varios registros (y no sólo uno) de Juan López. Estos registros muestran información histórica acerca de Juan López. Por ejemplo para descubrir donde ha vivido Juan López el último año, estos registros se buscan en el Data Warehouse. No hay coincidencia entre los registros en el ambiente operacional y el ambiente Data Warehouse. Si hay un cambio en la dirección de Juan López, entonces un nuevo registro es creado en el Data Warehouse reflejando de dónde a dónde cambio de dirección Juan López, debemos notar que los registros en Data Warehouse no son coincidentes, además que algunos elementos de tiempo son asociados a cada registro dentro del Data Warehouse.

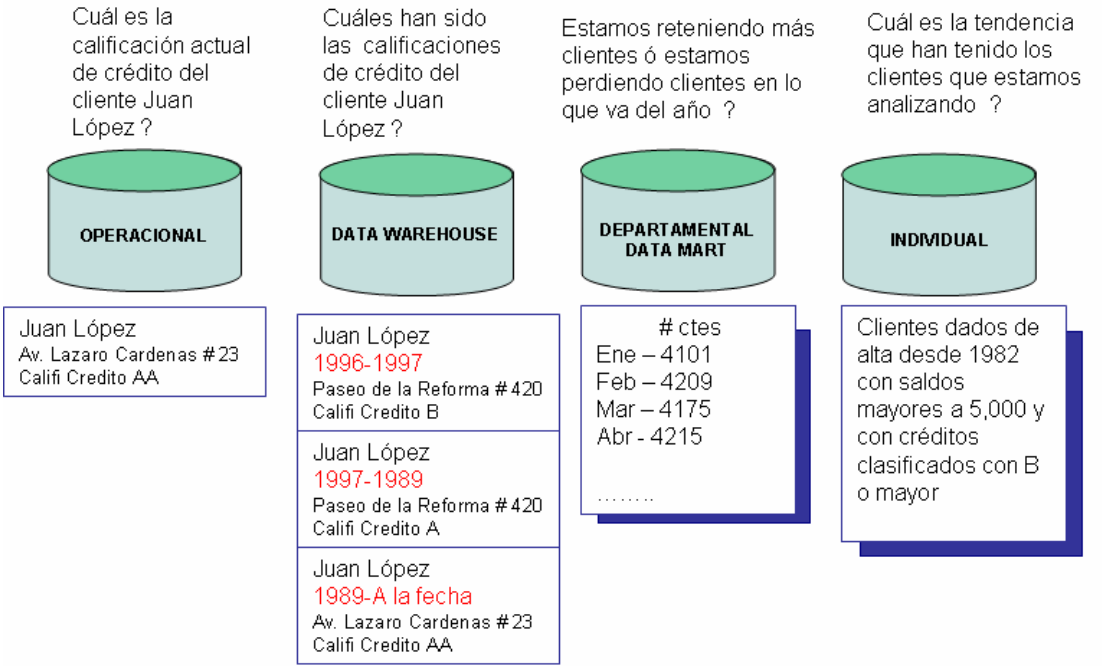
El ambiente Departamental contiene información local útil para diferentes departamentos dentro de la organización. Por ejemplo para el departamento de Marketing, Contabilidad, Riesgos, Manufactura u Operaciones, etc. El ambiente departamental en ocasiones es llamado "datamart[9]", OLAP[10] (On-line Analog Process) ó Sistemas de gestión de Base de Datos Multidimensional ("multidimensional DBMS").

[9] **DataMart** : Conjunto de hechos y datos organizados para soporte de decisiones en la necesidad de un área o departamento especifico. Los datos son orientados a satisfacer las necesidades particulares o locales de un departamento dado teniendo sólo sentido para el personal de ese departamento y sus datos no tienen porque tener las mismas fuentes que otros DataMarts.
[10] **OLAP** : Conjunto de principios que proveen un ambiente de trabajo multidimensional para el soporte de toma decisiones.

El típico dato del ambiente departamental es un archivo a nivel cliente mensual. En el archivo se encuentra una lista de clientes por categoría el detalle del cliente Juan López se pierde ya que va sumado en la cantidad mensual de clientes.

El último nivel de datos es el ambiente individual. Los datos en este ambiente son temporales y de volúmenes pequeños. En este ambiente se realiza análisis con consultas no planeadas, es decir, análisis que requieren varias aproximaciones para llegar a un resultado. Dichas aproximaciones dependerán del analista que esté realizando el estudio o reporte. Una regla para el ambiente individual es que los datos están pensados para ser soportados por una PC. Los reportes con información ejecutiva ó directiva (EIS executive information system) son típicamente procesados en este ambiente.

Figura 2.4 UN EJEMPLO SIMPLE



Por último en este marco teórico se describirán los CONCEPTOS DE MODELOS DE DATOS MULTIDIMENSIONAL .

Para construir una base de datos multidimensional, se debe iniciar con un modelo de datos multidimensional. El modelo de datos multidimensional provee un método para implementar una base de datos sencilla y entendible. Una base de datos multidimensional se puede ver como un cubo de tres o cuatro dimensiones donde los usuarios pueden acceder a parte de la sección de la base de datos a través de alguna de sus dimensiones. Para crear una base de datos multidimensional se requiere un modelo que permita visualizar los datos.

Pongamos un ejemplo clásico en cualquier empresa donde se quiere evaluar el rendimiento de las ventas por producto en el tiempo. Es fácil visualizar el proceso del negocio como un cubo de datos, el cual contiene dimensiones por tiempo, productos y mercado.

La siguiente figura muestra este modelo dimensional. Todas las intersecciones a lo largo de las líneas del cubo pueden contener las medidas o métricas del negocio. La medida ó métrica corresponde a un dato en una particular combinación de producto, mercado y tiempo.

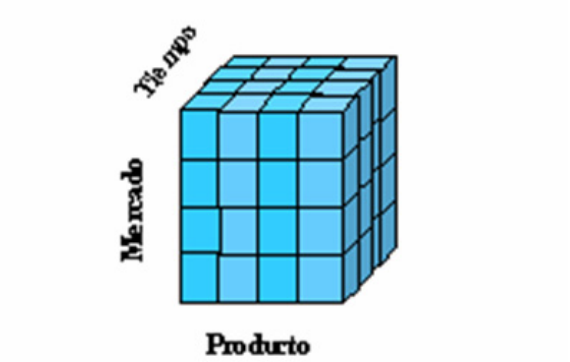


Figura 2.5 Un modelo dimensional de un Negocio que tiene dimensiones de Tiempo, Producto y Mercado

Otro nombre para el modelo dimensional es "diagrama estrella". El diseño de base de datos usa este nombre porque el diagrama para este modelo luce como una estrella con una tabla central y alrededor de esta un conjunto de otras tablas. La tabla central es la única tabla con múltiples *joins* conectados con el resto de las otras tablas. La tabla central es nombrada como "Fact Table (Tabla de Hechos)" y las otras tablas a su alrededor son "tablas de dimensiones". Todas las tablas de dimensiones tienen un *join* simple que se relaciona con la Fact Table. La siguiente figura muestra un modelo multidimensional simple:

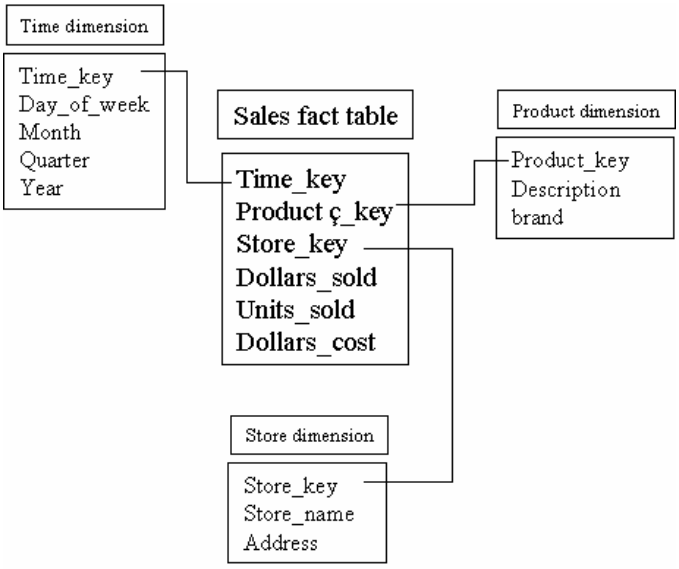


Figura 2.6 Representación de un modelo estrella

La *tabla Fact* o tabla de Hechos (Fact Table).

La tabla de hechos almacena las medidas ó métricas del negocio y puntualiza los valores de las llaves que a su vez serán el nivel mínimo de cada tabla de dimensiones. Las medidas ó métricas son cualitativas ó basadas en los objetivos del tema de negocio a representar. Las medidas ó métricas son generalmente numéricas y responden a aspectos de preguntas como ¿cuánto?, ejemplos de métricas son el precio, el importe de ventas, gastos, ingresos, etc. Las métricas pueden estar basadas en una columna en una tabla ó pueden ser calculadas.

Antes de diseñar una tabla de hechos, se debe determinar su granularidad. La granularidad es como se define el nivel más bajo de un registro en la tabla. Por ejemplo, la tabla de hechos que se muestra enseguida contiene un registro por cada producto vendido a cada cuenta por cada día.

Código del Producto	Código de la cuenta	Código del Día	# de unidades vendidas	Importe de los ingresos	Importe la rentabilidad
1	54321	19980501	1	82.12	27.12
3	42560	19980510	2	171.12	66
1	32344	19980521	1	82.12	27.12

Tabla 2.2 Ejemplo de una tabla de hechos

Dimensiones del Modelo de Datos

Una dimensión representa un sencillo conjunto de objetos o eventos en el mundo real. Cada dimensión que se identifique para el modelo de datos será una tabla de dimensión. Las dimensiones son la calificación que se hará a las métricas de la tabla de hechos, principalmente porque las dimensiones responden el Qué, Cómo, Cuándo y Dónde de los aspectos de una pregunta. Si consideramos las siguientes preguntas de negocio, la dimensión se identifica por el subrayado:

¿Qué cuentas produjeron el más alto ingreso el año pasado?

¿Cuál fue nuestra rentabilidad por ejecutivo?

¿Cuántas unidades se vendieron por producto?

Elementos de las dimensiones

Una dimensión puede definir múltiples elementos de dimensión para diferentes niveles de resumen. Por ejemplo, todos los elementos que relacionan a la estructura de ventas de una organización comprenden una dimensión. La siguiente figura muestra los elementos de dimensiones de la dimensión estructura.

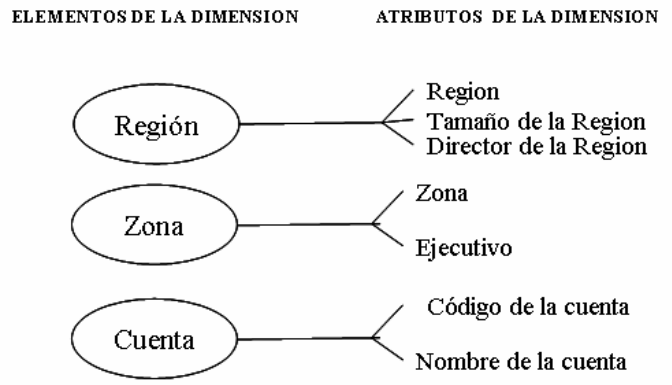


Figura 2.7 Elementos y Atributos de una dimensión

Las dimensiones están formadas por jerarquías que relacionan los elementos. Gracias al aspecto jerárquico de las dimensiones el usuario puede accesar a los datos de un nivel alto a un nivel bajo de detalle. La figura anterior muestra la relación de la jerarquía con los elementos de la dimensión: De las cuentas puede pasar a la zona y de la zona a la región. Los usuarios pueden consultar diferentes niveles de la dimensión, dependiendo de los datos que los usuarios quieran extraer.

Los elementos de dimensiones usualmente se almacenan como códigos numéricos o caracteres cortos para facilitar los *joins* con otras tablas; cada elemento de dimensión puede definir múltiples atributos de dimensión en la misma forma en que la dimensión puede definir múltiples elementos de dimensión.

Atributos de dimensión.

Un atributo de dimensión es una columna en una tabla de dimensión. Cada atributo describe un nivel de agrupación dentro de la jerarquía de dimensión. Los elementos de dimensión definen la jerarquía, los atributos describen a los elementos en términos de que sea familiar para los usuarios. La figura anterior muestra los elementos de dimensión y los atributos correspondientes. Como los atributos describen los elementos en una dimensión es más usual que los atributos sean texto.

Tablas de dimensiones.

Una tabla de dimensión es una tabla que almacena la descripción textual de las dimensiones de negocio. Una tabla de dimensiones contiene elementos y atributos apropiados para cada nivel en la jerarquía. El nivel más bajo de detalle que es requerido en el análisis de datos lo determina el nivel mas bajo en la jerarquía. Los niveles mas altos son la base del nivel de redundancia de datos. Esta desnormalización en las tablas reduce el número de joins que son requeridos para la consulta (query) y hace más fácil al usuario la consulta de los niveles altos y haciendo *drill down* baja a los niveles bajos de detalle. El termino *drilling down* significa bajar de nivel desde las jerarquía de las tablas de dimensiones.

Diseño de un modelo multidimensional.

Para la construcción de un modelo multidimensional es necesaria una metodología que permita desarrollar el diseño completo de base de datos. Una importante tarea en el diseño de la base de datos es iniciar con las fuentes existentes de datos que la

organización usa. Una vez que los procesos son identificados, una ó más tablas de hechos son construidas para cada proceso de negocio. Los siguientes pasos describen la metodología que se emplea para la contracción del modelo de datos.

Para diseñar una base de datos multidimensional.

1. Elegir el (los) proceso(s) de negocio que se desean usar para el tema de análisis que se desea modelar.
2. Determinar la granularidad de la tabla(s) de hechos
3. Identificar dimensiones y jerarquías de cada tabla de hechos
4. Identificar las métricas para la(s) tabla(s) de hechos
5. Obtener el visto bueno de los usuarios del modelo de datos

1. Elegir el (los) proceso(s) de negocio.

Un proceso de negocio es una importante operación dentro de la organización que soportan algunos sistemas origen. Se coleccionan datos desde este sistema origen para usuarios en la base de datos multidimensional. Los procesos de negocio identifican de donde vienen los datos, y como deben transformarse los datos para convertirse en información útil.

2. Determinar la granularidad de la tabla de hechos

Una vez que se ha reunido toda la información relevante acerca de tema de análisis, el siguiente paso en el diseño del proceso es determinar la granularidad de la tabla de fact. Para hacer esto es necesario decidir cual será el nivel más bajo que contendrá la tabla de fact. La granularidad corresponderá directamente a las dimensiones del modelo de datos. De este modo cuando se defina la granularidad de la tabla de hechos, se identificaran las dimensiones del modelo de datos.

La granularidad también determinara cuanto espacio es requerido para el almacenamiento de la base de datos.

3. Identificar las dimensiones y jerarquías

Una vez que se definió la granularidad de la tabla de hechos, es muy sencillo identificar las principales dimensiones para el modelo de datos porque cada componente que define la granularidad corresponde a una dimensión. Enseguida se deberán mapear los elementos y jerarquías para cada dimensión. Es importante considerar las preguntas que responderá el modelo de datos para definir la granularidad. Este paso también incluye determinar los atributos para cada tabla de dimensiones. Una vez que la tabla de hechos se ha definido, se puede decidir cuales serán los atributos de las dimensiones para cada tabla de dimensión. Se asignara uno ó mas atributos a cada elemento de dimensión para describirlos; ejemplo: el elemento día (fecha) se describirá con los nombre de los días Lunes, Martes, Miércoles, etc.

4. Identificar las métricas para la(s) tabla(s) de hechos

Las métricas para el modelo de datos no solo incluyen datos por si mismos, también pueden definirse nuevos valores que se calcularan de datos ya existentes. Cuando se examinan las métricas, en ocasiones se descubre que es necesario hacer ajustes a la

definición de la granularidad de la tabla de fact y las dimensiones. Para la definición es importante responder la pregunta: ¿qué métricas son usadas para el análisis del negocio?

5. Obtener el Visto Bueno de los usuarios del modelo de datos

Simplemente asegurarse que el usuario esta satisfecho con los datos resultado del modelo.

3.OBJETIVO DE NEGOCIO: Modelo de Rentabilidad

En junio de 1998, en Banco BITAL se decidió a comenzar con una nueva fase llamada "Modelo de rentabilidad" que consideraba la implementación de los procesos de rentabilidad que permitía evaluar el desempeño de las sucursales a través de toda la estructura del banco llegando incluso al cliente para calificarlo en función de su rentabilidad. El proyecto contemplaba la entrega de una aplicación que tuviera la funcionalidad de un Tablero de Control[11] que sería publicada a todas las sucursales a nivel nacional a través de la Intranet.

El proyecto "Modelo de Rentabilidad" se impulsó para justificar los costos y beneficios que traería la tecnología de un Data Warehouse al Banco, y de igual manera este modelo vendría a ayudar a la gestión y conocimiento del los clientes del Banco. La descripción de la necesidad del proyecto ayudó a entender porqué fue necesaria una estrategia de Data Warehouse para su solución.

DESCRIPCIÓN DE NEGOCIO DEL MODELO DE RENTABILIDAD

A la dirección le interesaba conocer a los clientes y calcular el valor de cada uno de ellos. Las siguientes métricas se autorizaron para iniciar el proyecto:

ESPECIFICACIONES DE LA DIRECCIÓN	
Tiempo invertido	4 años
Fecha de inicio proyecto	1998
Objetivos de negocio	Unificación de clientes y limpieza de datos que permitan un calculo adecuado de rentabilidad por cliente
	Incrementar 15% la utilidad de clientes rentables (A,B,C)
	Incrementar ingresos por venta cruzada
	Retención de clientes
Ingresos esperados	US$120 millones anualmente (considerando el 1er. Y 2do. obj.)
Fecha final del objetivo	Final 2002
Estrategia	Unificar fuentes de información

[11] Tablero de Control: Herramienta de medición de las actividades y resultados diarios de la empresa. De fácil acceso que permite al usuario medir, controlar, administrar, tomar decisiones y dirigir sus unidades de negocio, por lo regular se distribuyen por la intranet o Internet de la empresa mediante aplicaciones WEB.

	Construcción del DWH utilizando personal de Bital. Consultores sólo para resolver casos específicos.
	Construcción de un modelo de Rentabilidad
Tema del Data Warehouse	Rentabilidad a nivel cliente y cuenta

Tabla 3.1 Requerimientos de la Dirección

Contando con el apoyo de la Dirección, se definió el equipo de usuarios que estaría trabajando y definiendo las reglas de negocio, aunque no eran los únicos usuarios al menos recolectaba el 80% de las necesidades, este equipo estaba conformado por el área Control e Información Directiva (esta área se creó en noviembre de 1998 para la generación del Presupuesto de Banca Comercial, desde marzo de 1999 se dedicó a crear los cimientos de la arquitectura de información, integrando la información del negocio y diseñando reportes de resultados que se envían a los distintos niveles jerárquicos, desde la Dirección hasta las Unidades de Negocio, con el fin de que se puedan tomar decisiones con base en la conformación de saldos de captación y de colocación, cara a los productos, los clientes y su comportamiento) el alcance del proyecto fue definido por esta área de la siguiente manera :

Generar con periodicidad Mensual el siguiente calculo

- Calculo de rentabilidad de las cuentas de Ahorro e Inversión (Captación)
- Calculo de rentabilidad de los créditos y tarjetas de crédito (Colocación)
- Calculo de rentabilidad de los contratos Bursátiles y de Casa de Bolsa (Captación No tradicional)

_____

= CALCULO DE RENTABLIDAD DE CADA CLIENTE

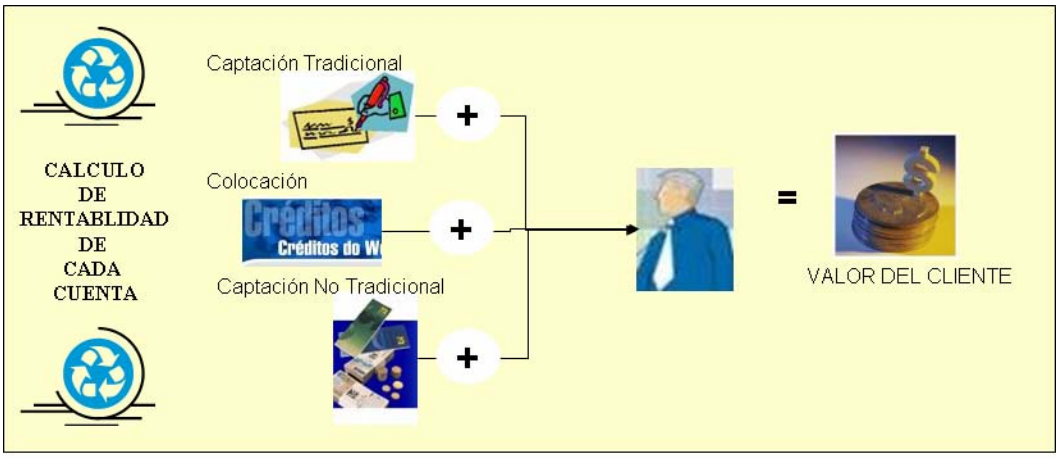


Figura 3.1 Definición del proceso de rentabilidad

Donde el cálculo de rentabilidad esta dado por la siguiente fórmula:

	INTERÉS COBRADO
-	**INTERÉS PAGADO**
=	**MARGEN FINANCIERO**
+	**INGRESOS DE LA OPERACIÓN**
-	**EGRESOS DE LA OPERACIÓN**
-	**GASTOS DE ADMINISTRACIÓN Y PROMOCIÓN**
=	**UTILIDAD**

Tabla 3.2 Fórmula de Rentabilidad

INTERÉS COBRADO (ò Ingresos por Intereses) : Se consideran como ingresos por intereses los rendimientos generados por la cartera de crédito, contractualmente denominadas como intereses, así como los premios e intereses de otras operaciones financieras propias de las instituciones financieras tales como: Depósitos en Instituciones de Crédito, inversiones en valores, operaciones de reporto y préstamos de valores.

INTERÉS PAGADO (ó Gastos por Intereses): Se consideran gastos por intereses los premios e intereses derivados de la captación de las instituciones, incluidos los relativos a operaciones de reporto y préstamos de valores, así como intereses y primas relativos a las obligaciones subordinadas.También se consideran como Gastos por intereses las comisiones a cargo derivadas de préstamos recibidos por la institución o la colocación de deuda bancaria.

MARGEN FINANCIERO : El margen financiero debe estar conformado por la diferencia entre los ingresos por intereses y los gastos por intereses, incrementados o deducidos por el resultado por posición monetaria relacionado con partidas del margen financiero.

Como INGRESOS DE LA OPERACIÓN se consideran:

Resultado por valuación a valor razonable de valores, títulos en reporto, divisas y metales preciosos.

Comisiones por prestación de Servicios de Administración o custodia de recursos.

Comisiones por uso o emisión de tarjeta de crédito.

Resultado por compra-venta de valores, instrumentos financieros, derivados, divisas y metales preciosos amonedados.

Como EGRESOS DE LA OPERACIÓN se consideran:

Resultados por valuación a valor razonable de divisas, títulos en reporto, divisas y metales preciosos.

Resultado por compra venta de valores, instrumentos financieros derivados, divisas y metales preciosos.

Dentro de GASTOS DE ADMINISTRACIÓN Y PROMOCIÓN deben incluirse: remuneración al personal y consejeros, prestaciones al personal y consejeros, honorarios, rentas, gastos de promoción, gastos no deducibles, depreciaciones y amortizaciones,

UTILIDAD (ó Resultado de la Operación): corresponde a los ingresos (egresos) totales de la operación, disminuidos por los gastos de administración y promoción de la institución.

La información debería mostrarse a todos los niveles de la estructura Dirección General hasta llegar al ejecutivo y la sucursal, permitiendo ver el detalle del cliente y su cartera que originó la rentabilidad. Esta estructura permite evaluar la rentabilidad en todos los niveles. Los niveles en esta estructura son los siguientes:

Figura 3.2 Niveles de agrupación requeridos

Las vistas de la información requerida fueron las siguientes:

Para todos los niveles se requería un acceso amigable por WEB que permitiera evaluar el estado de resultado.

Un ambiente que permitiera realizar análisis para encontrar tendencias y comportamientos de los clientes, oportunidades y amenazas en las unidades de negocio, etc. Dirigido a un grupo reducido de personas en el Banco.

Tanto para presentar la información en Reportes fijos en WEB como para el ambiente de análisis la Historia requerida fue de un año.

A continuación se describen brevemente los sistemas involucrados en el proceso de rentabilidad, que se identificaron de acuerdo a las variables de rentabilidad definidas dentro de este alcance.

Sistema	Descripción
Sistema de Clientes	Sistema de administración y almacenamiento de perfiles de clientes y su relación con el banco.
Sistemas de Ctas. de Cheques	Sistema transaccional de administración y almacenamiento para las cuentas de depósito.
Sistema de Créditos	Sistema transaccional de administración y almacenamiento para los créditos
Sistemas de Tarjetas de Crédito	Sistema transaccional de administración y almacenamiento para el área de colocación y tarjeta de crédito.
Sistema Bursátil	Sistema transaccional de administración y almacenamiento para los contratos Bursátiles propios del Banco
Sistema de Casa de Bolsa	Sistema transaccional de administración y almacenamiento para los contratos bursátiles de la filial Casa de Bolsa
Sistema de Recursos Humanos	Sistema transaccional de administración y almacenamiento para el control de la nómina de BITAL.
Sistema de Costos	Sistema transaccional de administración y almacenamiento para los gastos de infraestructura.
Archivos con información de comisiones	Archivos de datos mayormente en Excel que se llevan para el registro de comisiones principalmente

Tabla 3.3 Sistemas Operacionales involucrados en el proyecto

El diagrama muestra los flujos de información que se identificó para el proceso de Rentabilidad:

Figura 3.3 Flujos de información del modelo de rentabilidad

Los sistemas antes mencionados tienen plataformas muy variadas tales como:

Sistemas Operativos: MVS, UNIX, Windows

Hardware: IBM, SUN, Tandem

DBMS: Progress, Informix, VSAM, DB2, Datacom

Además de considerar la problemática de los problemas en la variedad de plataforma para recolectar los datos, el volumen en la cantidad de datos representó un reto para el procesamiento de los datos; la siguiente tabla muestra los volúmenes por sistema origen representado en número de registros en un periodo mensual.

SISTEMA	NIVEL MÍNIMO DE INFORMACIÓN	NÚMERO DE REGISTROS
Sistema de Clientes	Cliente – Cuenta	11,000,000
Sistemas de Cuentas de Cheques	Transacción – Cuenta-Cliente	219,600,000
Sistema de Créditos	Transacción – Cuenta-Cliente	50,000,000
Sistemas de Tarjetas de Crédito	Transacción – Cuenta-Cliente	50,000,000
Sistema Bursátil	Transacción – Cuenta-Cliente	500,000
Sistema de Casa de Bolsa	Transacción – Cuenta-Cliente	500,000
Sistema de Recursos Humanos	Empleado	20,000
Sistema de Costos	Departamento	100,000
Archivos con información de comisiones	Varios niveles	20,000

Tabla 3.4 Volúmenes de los Sistemas Operacionales

Con la solución propuesta de crear una estructura de Data Warehouse, se inicio en identificar y trabajar en preparar un ambiente que soportara la transición de una estructura con sistemas transaccionales a una estructura de sistemas de soporte a decisiones.

4. ESTRATEGIA DE SOLUCIÓN

Con el inicio de este proyecto, el banco se incorpora a la estrategia de tener un data warehouse corporativo para dar solución a sus problemas de integración, accesibilidad y calidad de información, como lo hacían otras instituciones financieras en México persiguiendo los siguientes objetivos:

- Proveer de una sola versión de la realidad
- Proveer a los usuarios de una sola fuente confiable de información
- Proveer una sola fuente común para todos los usuarios además de ser la única fuente de acceso.
- Otorgar un acceso fácil a la fuente de información integrada
- Proveer de una perspectiva de historia de la información
- Otorgar calidad en los datos
- Eliminar varios reportes con la misma información en diferentes áreas
- Minimizar los requerimientos a sistemas para nuevos reportes.

La definición de DATA WAREHOUSE que se adopto fue: Es un proceso que agrupa datos desde múltiples fuentes, incluyendo datos históricos para soportar la continua necesidad de consultas, reportes analíticos y soporte de decisiones y con la gran ventaja que es separado, es decir, que no interfiere con los sistemas operativos de actividad crítica en el Banco. Considerando que el DATA WAREHOUSE no es ni un producto de software ni una máquina, o tecnología de bases de datos en particular, sino una serie de componentes y procesos que en conjunto forman la arquitectura llamada DATA WAREHOUSE.

Como ya se había comentado, no fue este proyecto "Modelo de Rentabilidad" el primero en proponer esta arquitectura ya el Banco había apostado en un Data Warehouse con un proyecto anterior poco exitoso que se desarrolló de 1995 a 1996, que buscaba integrar una base de datos de clientes orientada a satisfacer las necesidades de información de las áreas de Mercadotecnia, a partir de los datos almacenados en los diferentes sistemas que soportaban la operación de los productos del Banco, pero por la poca experiencia en el procesamiento de datos masivos no se obtuvo ningún resultado. El diseño de modelo de rentabilidad aprovecho lo adquirido por el proyecto anterior y se baso en la siguiente infraestructura:

- ✓ Plataforma: Sun 10,000
- ✓ Sistema Operativo: Unix Solaris 2.6
- ✓ DBMS : Se instalo Informix On Line pero poco después se empleó Informix XPS
- ✓ Leguajes de Programación: Se inició utilizando Informix 4gl y enseguida se utilizó Esql-c, Shell y cron-shell.

Se armo un grupo de trabajo conformado por:
6 usuarios de negocio.
4 diseñadores de los modelos de datos (incluyendo un experto en la Base de Datos informix)
3 programadores (Esql-c y shell's)
2 programadores web

Y el apoyo de áreas de soportes técnico con funciones ambientar y configurar el equipo.

En las siguientes imágenes se muestra la estrategia de solución que se diseñó donde se parte de la problemática a resolver hasta llegar a una estructura de Data Warehouse donde vivirá una estructura de datos multidimensional con el modelo de Rentabilidad.

La Problemática a resolver cara al usuario era reducir los tiempos de entrega de información y romper con el proceso deficiente de obtención de información.

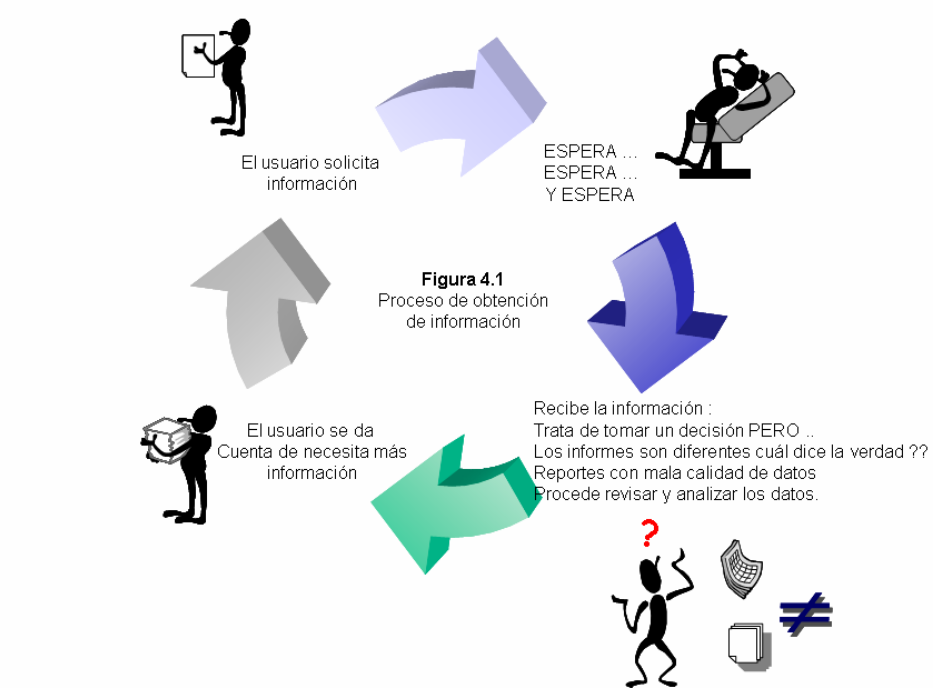


Figura 4.1
Proceso de obtención de información

La problemática se origina por la estructura que soporta la obtención de información como se muestra la siguiente lámina.

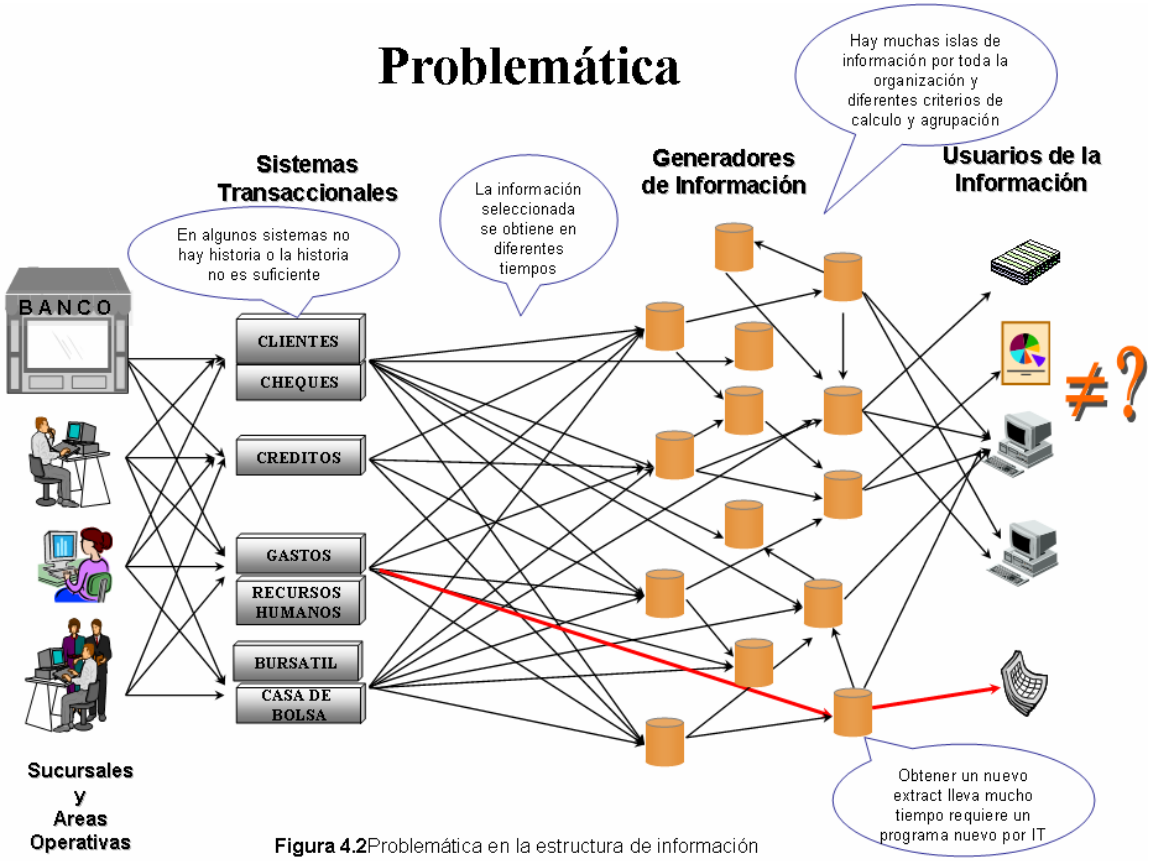


Figura 4.2 Problemática en la estructura de información

Existía una telaraña de islas de información a lo largo de la organización, que presentaban problemas como:

1) Problemas de Historia (ej. El sistema de Cheques sólo tenía información de un mes cuando era necesario más información se obtenía la información de otra fuente sin verificar si la información era confiable).
2) Selección de información en diferentes tiempos (ej. Para reportar cifras al cierre de mes un departamento tomaba información el día 3ro hábil, es decir, cifras previas al cierre definitivo mientras otro departamento tomaba cifras el día 8vo hábil con el cierre definitivo).
3) Diferentes criterios de agrupación y calculo, cada área localmente define sus reglas.
4) Si el departamento generador de información no contaba con toda la información necesaria en su pequeño repositorio de datos el tiempo para que la gente de IT lo atendiera para extraer nuevos datos era muy largo.

La primera para de la estrategia de solución fue eliminar las islas de información y almacenar toda la información necesaria en un solo repositorio. El grupo de trabajo de Diseño estaba convencido que el mejor sistema fuente para el data warehouse era el ODS[12]. Típicamente el ODS es una base datos relacional con estructura normalizada

[12] **ODS (operational data store):** repositorio consistente de información que sirve a los fines de brindar soporte a la toma de decisiones operativas y rutinarias. Contiene información organizada por áreas de interés, pero con valores corrientes y no históricos

(para detalle del tema Base de datos Relacional ver Apéndice 1). Es aquí donde se coleccionarían e integrarían los datos y se asegurará que los datos estuvieran completos y actualizados; se definió que la base de datos contendría datos extraídos cada noche desde los sistemas operacionales.

Los beneficios buscados eran los siguientes:

1) Trasladar el procesamiento de información (entendiendo como procesamiento de información armar reportes, crear pantallas, extracciones, etc.) fuera del ambiente productivo del sistema operacional. Es muy natural que el procesamiento de información esté en constante cambio: las condiciones de negocio cambian, la organización cambia, las prácticas de contabilidad cambian, etc. Cada uno de estos cambios tiene un efecto sobre el proceso de agrupación de la información ó en el proceso de generar información. Cuando el procesamiento de datos se incluye en el ambiente productivo de los sistemas operacionales, esto dará la apariencia que habrá una proceso de mantenimiento eterno. Al mover el procesamiento de información fuera del ambiente productivo y pasarlo al ambiente Data Warehouse la carga de mantenimientos se reducirá considerablemente.

2) Otro importante efecto al trasladar una gran cantidad de datos que la mayoría de las veces estaban almacenados o archivados en los ambientes productivos de los sistemas operacionales, se liberan recursos que tienen que ver con las siguientes tareas:

a) Almacenamiento
b) Correcciones
b) El monitoreo
c) Creación de índices

Dando el beneficio de que serán más fáciles de utilizar en el ambiente Data Warehouse que en ambiente productivo de los sistemas operacionales.

3) Se definirán extracts únicos con fotos de información actual, por lo que se reducirá considerablemente las peticiones al área de IT y se resolverá el problema del tiempo en que se toman los datos, así como, el problema de historia se resolverá al almacenar y acumular los datos en nuevo repositorio fuera del ambiente productivo de los sistemas operacionales.

La siguiente imagen muestra la estrategia de solución inicial a seguir:

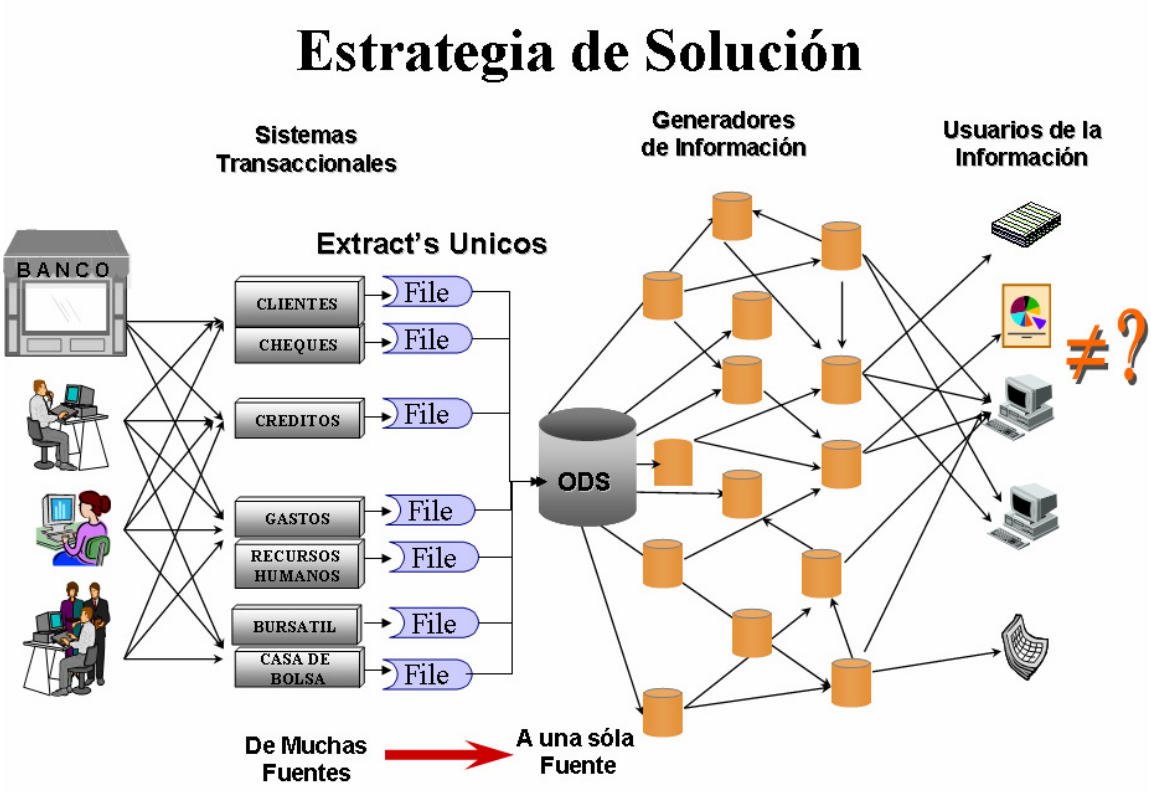


Figura 4.3 Estrategia de solución con un ODS

El siguiente paso en la estrategia de solución fue planear la construcción del modelo multidimensional de Rentabilidad, que daría respuestas a las preguntas de negocio más comunes. Siendo Informix el gestor de Base de Datos, se decidió seguir su metodología para la creación del modelo multidimensional, siguiendo los pasos:

1. Elegir el (los) proceso(s) de negocio que se desean usar para el tema de análisis que se desea modelar.
2. Determinar la granularidad de la tabla(s) de hechos
3. Identificar dimensiones y jerarquías de cada tabla de hechos
4. Identificar las métricas para la(s) tabla(s) de hechos
5. Obtener el visto bueno de los usuarios del modelo de datos

Además este repositorio DWH tendría las siguiente características:

1. No habría transformaciones, cálculos, integraciones, agregaciones de información, estas se realizarían el repositorio ODS, mientras que éste repositorio solo realizaría cargas sencillas.
2. Este repositorio contendría la historia.
3. En su mayoría los datos de este repositorio serían datos derivados.

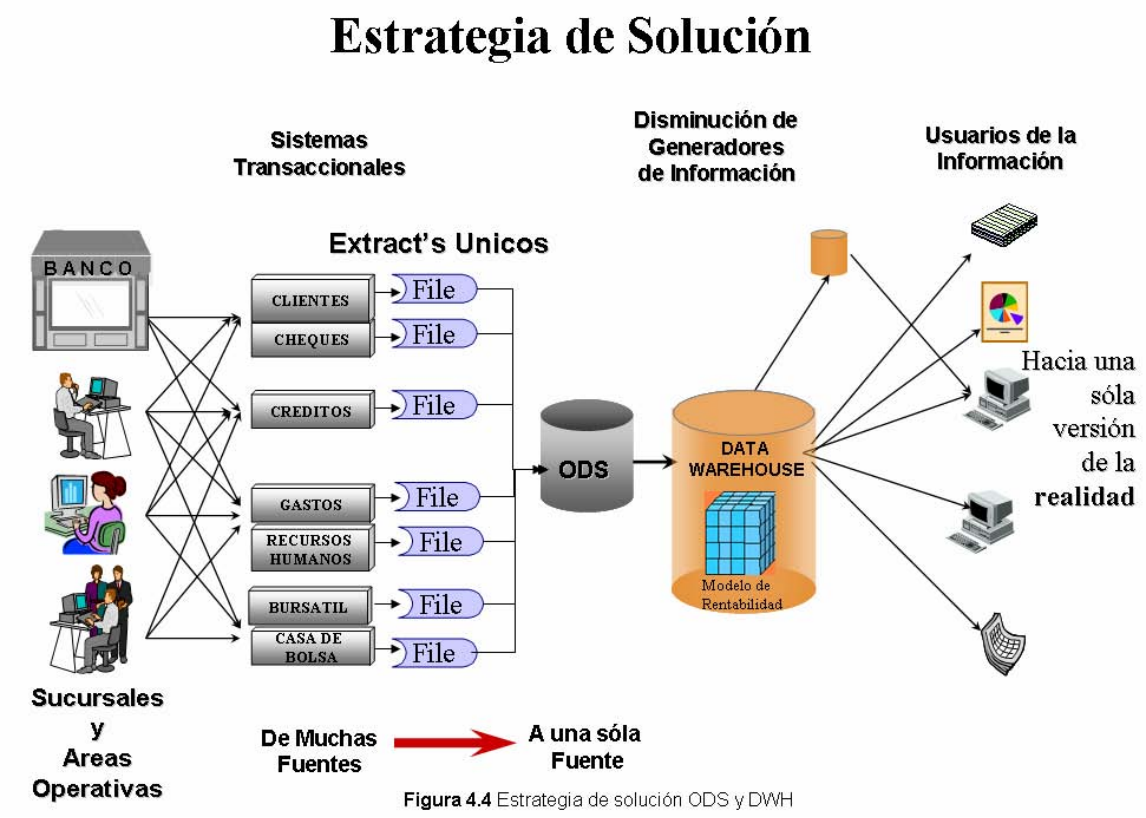


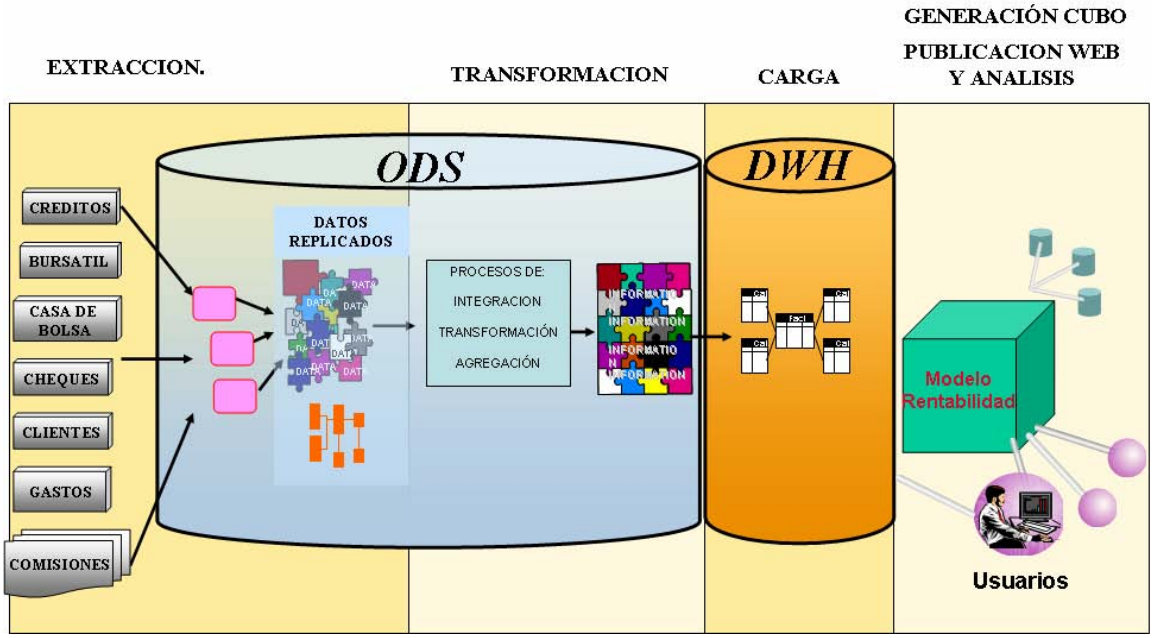
Figura 4.4 Estrategia de solución ODS y DWH

5. DISEÑO

DISEÑO DEL PROCESO ETL (EXTRACTION, TRANSFORMING AND LOADING)

Se diseñó un proceso que debería cubrir las etapas de ETL, generación de cubos, administración de herramientas de análisis y publicación Web, como muestra la siguiente figura:

Figura 5.1 Diseño del proceso DWH



Retomando el alcance descrito del proyecto, se tendría que generar:

- Calculo de rentabilidad de las cuentas de Ahorro e Inversión (Captación)
- Calculo de rentabilidad de los créditos y tarjetas de crédito (Colocación)
- Calculo de rentabilidad de los contratos Bursátiles y de Casa de Bolsa (Captación No tradicional)

= CALCULO DE RENTABLIDAD DE CADA CLIENTE

Se definieron las siguientes interfases (etapa de extracción), procesos (etapa de transformación) y tablas:

1) Captación Tradicional. Para esta interfase se definieron.

a) 2 extract's diarios. Es de notar que estos extracts fueron diarios debido a que el sistema origen no contaba con la información mensual requerida que se describe en el siguiente punto.

b) Un proceso de cálculo de saldos promedio y agrupado de transacciones mensual.

c) La creación de una tabla principal a nivel cuenta-cliente con la rentabilidad de las cuentas de Ahorro e Inversión:

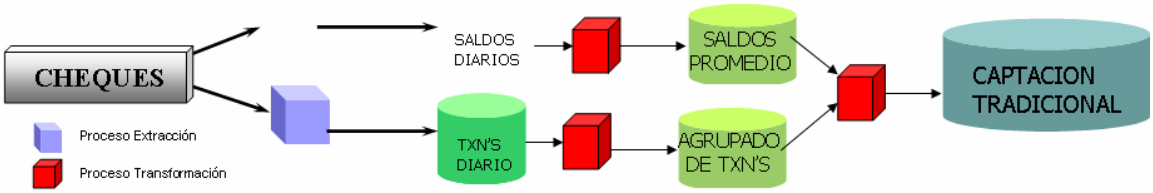


Figura 5.2 Diseño de interfase y proceso Captación Tradicional

2) Captación No Tradicional

a) 4 extract's mensuales de Transacciones y Saldos de los contratos bursátiles del Banco y de la Casa de Bolsa.

b) La creación de una tabla principal a nivel cuenta-cliente con la rentabilidad de los contratos Bursátiles y de Casa de Bolsa.

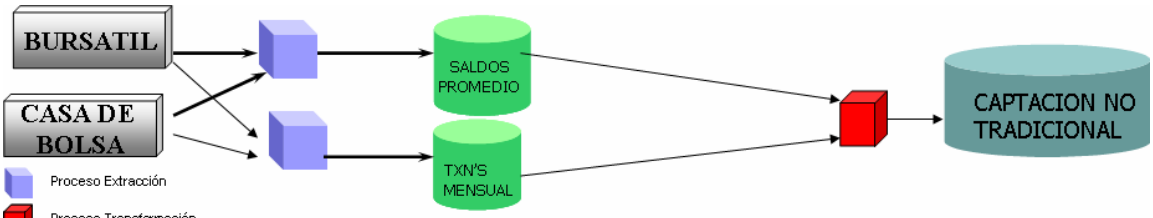


Figura 5.3 Diseño de interfase y proceso Captación No Tradicional

3) Colocación. Para esta interfase se definieron.

a) 4 extract's mensuales de Transacciones y Saldos y condiciones de los créditos y las tarjetas de crédito.

b) La creación de una tabla principal a nivel cuenta-cliente con la rentabilidad de las tarjetas de crédito y los créditos.

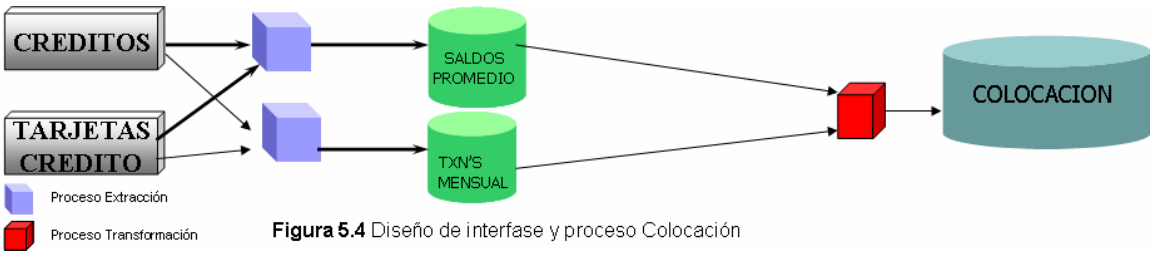


Figura 5.4 Diseño de interfase y proceso Colocación

4) Rentabilidad del cliente

a) Un proceso que integrará todas la cuentas de los diferentes productos en una sola tabla.

b) Un proceso que cargará la información de los clientes y que complementará la información de rentabilidad de los clientes, con datos como la edad, sexo, etc.

c) Un proceso de cálculo de la rentabilidad de cada cliente.

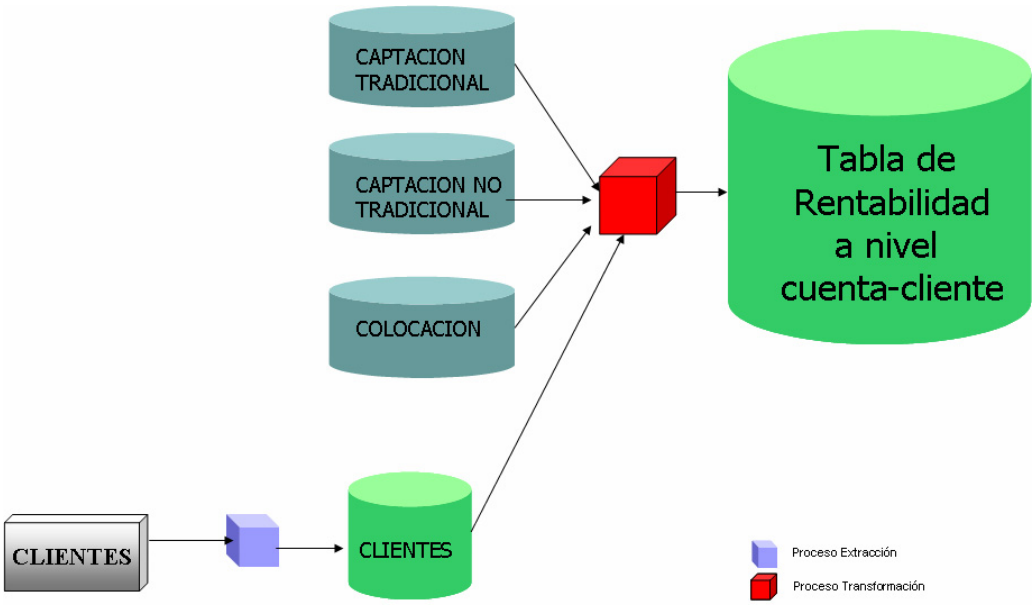


Figura 5.5 Diseño proceso Rentabilidad cta-cte

5) Modelo de rentabilidad

a) Se definieron los procesos de carga de los gastos y las comisiones.

b) Por último se diseñó el modelo multidimensional de Rentabilidad del Banco que sigue los pasos de la metodología de diseño de un modelo multidimensional de Informix.

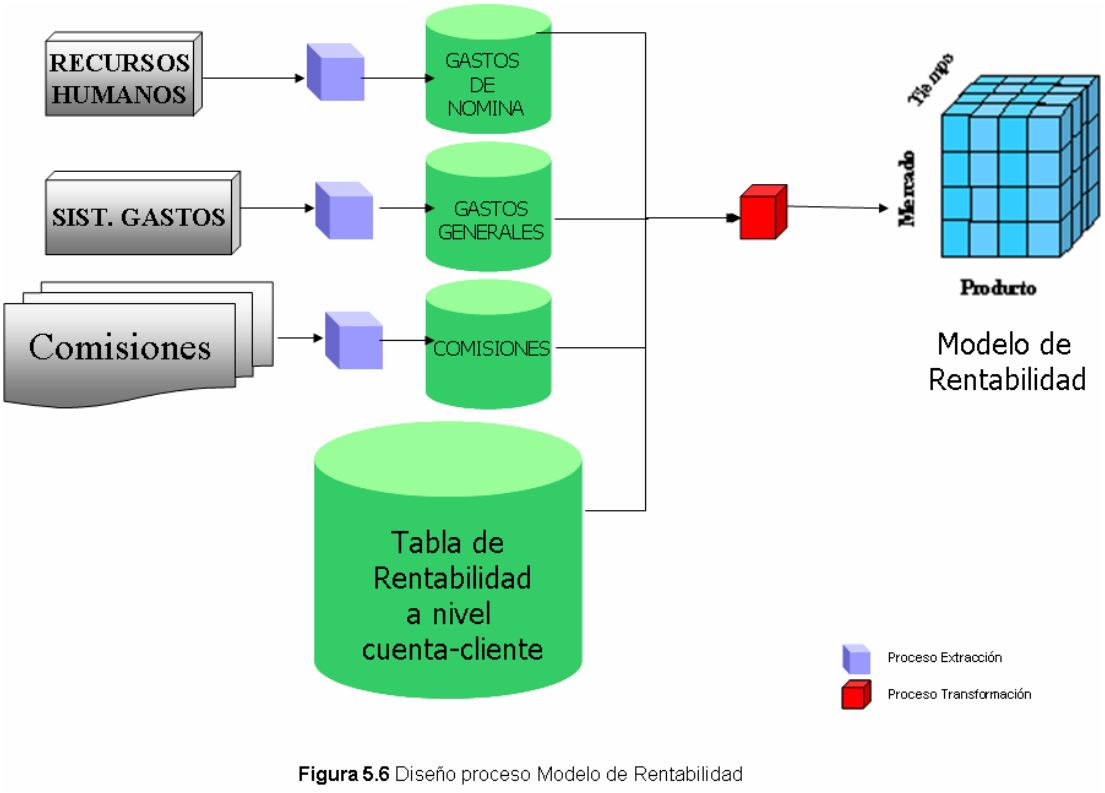


Figura 5.6 Diseño proceso Modelo de Rentabilidad

DISEÑO DEL MODELO MULTIDIMENSIONAL

El diseño del modelo de datos siguió los puntos listados a continuación:

1. **Elegir el (los) proceso(s) de negocio.** El proceso de negocio fue la Rentabilidad
2. **Determinar la granularidad de la tabla de hechos**

La granularidad estará definida por las siguientes llaves:

- ✓ cve_seg (PK)
- ✓ cve_prom (PK)
- ✓ cve_suc (PK)
- ✓ cve_mes (PK)
- ✓ id_prod (PK)

De tal manera que el nivel más bajo de la tabla de hechos sería segmento de banca por Promotor/Sucursal por Mes (Tiempo) y por Producto.

Cabe mencionar que la necesidad fue alcanzar el nivel cuenta, cliente pero esto sólo es una consulta informativa de datos al llegar al nivel de promotor; por lo que además se diseñó una tabla relacional que tenía la información relevante a nivel cuenta y cliente.

3. Identificar las dimensiones y jerarquías

Nombre de la Dimensión: Segmento de Banca.

Descripción: los clientes son atendidos por diferentes segmentos de banca de acuerdo a su perfil (si es persona moral, física, si es un empresario), de acuerdo a sus ingresos ó de acuerdo a los servicios que demanda. Cada segmento se especializa en atender a sus clientes para retenerlos y crecerlos ó para atraer nuevos clientes. A cada segmento se le piden ciertas metas y se les mide de acuerdo a su mercado. Enseguida se describen brevemente los grandes grupos de segmentos de banca, aunque cada uno agrupa un subconjunto de segmentos más específicos:

Banca Comercial o también llamada banca personal atiende a todos clientes que son personas físicas con ingresos medios.

Banca Patrimonial da atención personalizada a clientes preferentes con ingresos superiores a la media (personas físicas).

La Banca Privada provee a las personas que han consolidado un alto valor patrimonial y a sus familias, con las capacidades y el servicio personal que ellos necesitan para manejar eficazmente su riqueza

Banca Empresarial, todos los Productos y Servicios que ofrece para personas morales dentro de las cuales están las PYMES (Pequeñas y Medianas Empresas) ó personas físicas con actividad empresarial.

Banca Corporativa ofrece productos y servicios de mercado de capitales para los grandes corporativos.

Jerarquía, Elementos de la Dimensión y Atributos de la Dimensión: la siguiente tabla muestra la jerarquía, elementos de la dimensión y atributos de la dimensión. Los elementos de la dimensión se muestran en letra cursiva además que en el campo Nivel se identifican los niveles de agrupación siendo el nivel 1 el que contendrá mayor detalle de información. Los atributos de la dimensión se marcan con una "X" dentro de la columna atributo.

td_segmentos

Campo	Descripción	**Nivel**	Atributo
Cve_seg	Clave de Segmento de Banca	**1**	(FK)
Des_seg	Descripción del segmento banca		X
Gcve_seg	Clave de grupo de segmentos	**2**	
Des_gseg	Descripción del grupo de segmentos		X
Ggcve_seg	Clave de grupo de segmentos para presentar subtotales	**3**	
Des_ggseg	Descripción de la Clave de grupo de segmentos para presentar subtotales		X
gggcve_seg	Clave de grupo de segmentos para presentar cifras totales	**4**	
Des_gggseg	Descripción de la clave de grupo de segmentos para presentar cifras totales		X

Tabla 5.1 Tabla de la dimensión segmentos

Nombre de la Dimensión: Estructura

Descripción: esta dimensión muestra la representación de la estructura orgánica del banco, en este caso más específico sus unidades ó áreas de ventas en las que se muestran las relaciones que guardan entre sí los órganos que la componen. Esta dimensión facilita el entendimiento de las relaciones de jerarquía de las unidades de venta.

Los niveles de la estructura son los siguientes:

- ✓ Promotor
- ✓ Sucursal
- ✓ Subdirección ó Zona
- ✓ Dirección ó Coordinación
- ✓ Dirección Ejecutiva
- ✓ Dirección General

Jerarquía, Elementos de la Dimensión y Atributos de la Dimensión: La siguiente tabla muestra la jerarquía, elementos de la dimensión y atributos de la dimensión. Los elementos de la dimensión se muestran en letra cursiva además que en el campo Nivel se identifican los niveles de agrupación, siendo el nivel 1 el que contendrá mayor detalle de información. Los atributos de la dimensión se marcan con una "X" dentro de la columna atributo.

td_estructura

Campo	Descripción	**Nivel**	Atributo
cve_mes	**Clave de mes**	**1**	**FK**

cve_prom	**Clave de promotor**		
nom_prom	Nombre del promotor		X
reg_prom	Registro del promotor		X
cve_seg	Clave de Segmento		X
Estado	Estado donde labora		X
Cve_suc	**Clave de departamento de la Sucursal**	**2**	
Nom_suc	Nombre de la Sucursal		X
Reg_suc	Registro del responsable de la Sucursal		X
cve_coord	**Clave de departamento de la coordinación**	**3**	
nom_coord	Nombre de la coordinación		X
reg_coord	Registro del responsable de la coordinación l		X
cve_zona	**Clave de departamento la zona l**	**4**	
nom_zona	Nombre de la zona		X
reg_zona	Registro del responsable de la zona		X
cve_de	**Clave de departamento de la dirección ejecutiva**	**5**	
nom_de	Nombre dirección ejecutiva		X
reg_de	Registro de responsable dirección ejecutiva		X
cve_dg	**Clave de departamento de la dirección general**	**6**	
nom_dg	Nombre dirección general		X
reg_dg	Registro del responsable de la Dirección General		X

Tabla 5.2 Tabla de la dimensión estructura

Nombre de la Dimensión: Tiempo

Descripción: esta dimensión permitirá una compresión rápida de los periodos de tiempo en que se reportará la información, como en la tabla de hechos la unidad mínima de tiempo es el mes, la información se presentará en trimestres, cuatrimestres y años.

Jerarquía, Elementos de la Dimensión y Atributos de la Dimensión: La siguiente tabla muestra la jerarquía, elementos de la dimensión y atributos de la dimensión. Los elementos de la dimensión se muestran en letra cursiva además que en el campo Nivel se identifican los niveles de agrupación siendo el nivel 1 el que contendrá mayor detalle de información. Los atributos de la dimensión se marcan con una "X" dentro de la columna atributo.

td_tiempo

Campo	Descripción	**Nivel**	Atributo
cve_mes	**Clave de mes**	**1**	(FK)
desc_mes	Descripción del Mes		X
cve_trim	**Clave de trimestre**	**2**	
desc_cve_trim	Descripción del trimestre		X
cve_cuatri	**Clave de cuatrimestre**	**3**	
desc_cve_cuatri	Descripción del cuatrimestre		X
cve_anio	**Clave de año**	**4**	
desc_cve_anio	Descripción de año		X

Tabla 5.3 Tabla de la dimensión tiempo

Nombre de la Dimensión: Productos

Descripción: La dimensión Productos describe cada producto financiero que ofrece el Banco por las grandes categorías:

- ✓ Captación Tradicional
- ✓ Captación No Tradicional
- ✓ Colocación

Jerarquía, Elementos de la Dimensión y Atributos de la Dimensión: la siguiente tabla muestra la jerarquía, elementos de la dimensión y atributos de la dimensión. Los elementos de la dimensión se muestran en letra cursiva además que en el campo Nivel se identifican los niveles de agrupación siendo el nivel 1 el que contendrá mayor detalle de información. Los atributos de la dimensión se marcan con una "X" dentro de la columna atributo.

Td_productos

Campo	Descripción	**Nivel**	Atributo
id_prod	identificador del producto, subproducto	**1**	(FK)
desc_subprod	Descripción ó Nombre del subproducto		X
Cve_prod	Clave del producto	**2**	
desc_prod	Descripción ó Nombre del producto		X
Clase_prod	Categoría 1 de productos	**3**	
desc_clase_prod	Descripción ó nombre de la Categoría 1		X

Tabla 5.4 Tabla de la dimensión producto

4. Identificar las métricas para la(s) tabla(s) de hechos

Al responder la pregunta ¿qué métricas son usadas para el análisis del negocio?. Se definieron las siguientes:

Num_ctas_vig	Número de cuentas vigentes
Num_ctas_ven	Número de cuentas vencidas
Sdo_ven	Saldo vencido
Sdo_vig	Saldo vigente
int_cob	Interés Cobrado
int_pag	Interés Pagado
Mar_fin_prod	Margen Financiero
Comisiones	Comisiones
costo_trx	Costos transaccionales
Utilidad	Utilidad
Reservas	Reservas en ceros en este momento
tot_sueldo	Total de Gastos de Sueldo
tot_gralgas	Total de gastos generales

Tabla 5.5 Tabla con lista de métricas

Finalmente la tabla de hechos se definió de la siguiente forma:

Tabla de Fact:

Tf_renta_m

Cve_seg	Clave de segmento	Dimensión (PK)
Cve_prom	Clave de Promotor	Dimensión (PK)
Cve_suc	Clave de Sucursal	Dimensión (PK)
Cve_mes	Clave de mes	Dimensión

		(PK)
Id_prod	Identificador del producto, subproducto	Dimensión (PK)
Num_ctas_vig	Número de cuentas vigentes	Métrica
Num_ctas_ven	Número de cuentas vencidas	Métrica
Sdo_ven	Saldo vencido	Métrica
Sdo_vig	Saldo vigente	Métrica
Int_cob	Interés Cobrado	Métrica
Int_pag	Interés Pagado	Métrica
mar_fin_prod	Margen Financiero	Métrica
Comisiones	Comisiones	Métrica
costo_trx	Costos transaccionales	Métrica
Utilidad	Utilidad	Métrica
Reservas	Reservas en ceros en este momento	Métrica
Tot_sueldo	Total de Gastos de Sueldo	Métrica
Tot_gralgas	Total de gastos generales	Métrica

Tabla 5.6 Tabla Fact con dimensiones y métricas

Finalmente el modelo de datos multidimensional representado en un diagrama estrella que se muestra a continuación respondía las preguntas de Negocio más importantes en el eso momento:

¿QUÉ productos son más rentables?
¿QUIÉN vende mejor?
¿CUÁNDO son mejores las ventas (En qué temporadas del año)?
¿DÓNDE se vende mejor (En qué región) cierto producto?

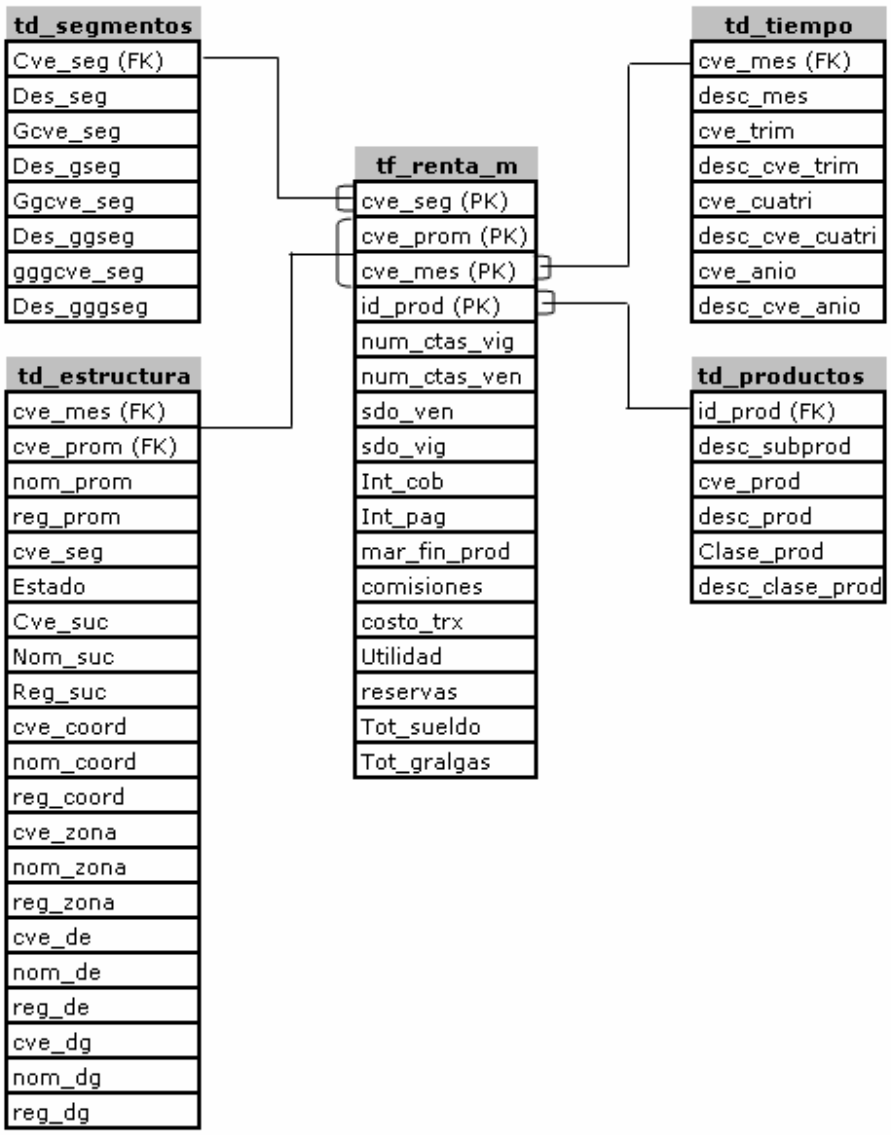


Figura 5.7 Modelo Estrella de Rentabilidad

5. Obtener el Visto Bueno de los usuarios del modelo de datos

El diseño se presentó a los principales usuarios:

- ✓ Usuarios de Negocio: Directores y subdirectores de Banca.
- ✓ Analistas de datos, Usuarios avanzados en explotar datos: Áreas staff encargadas de la elaboración de reportes a la Dirección
- ✓ Analistas denominados trabajadores de conocimientos: Áreas de Marketing

A cada uno de los usuarios se les ofreció la disponibilidad de los datos en diferentes clases de herramientas, de acuerdo a la siguiente pirámide.

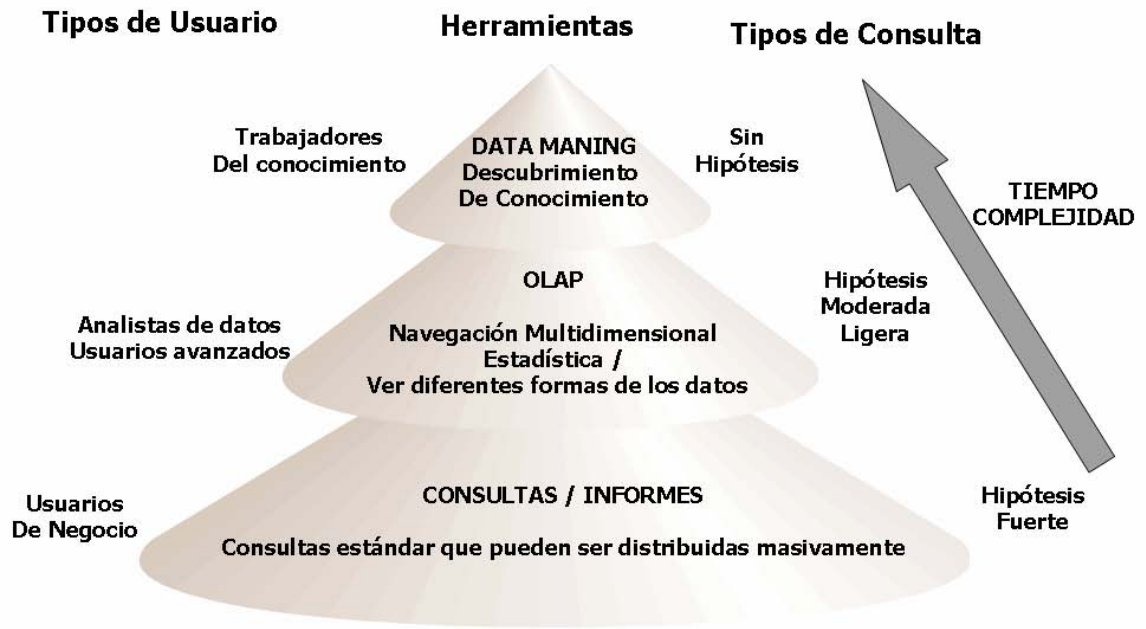


Figura 5.8 Pirámide de tipos de usuarios

Para los usuarios de Negocio, se ofreció distribuir la información relevante de rentabilidad a través de reportes en WEB dinámicos con consultas predefinidas.

Para los analistas de datos y usuarios avanzados: se ofreció la herramienta *Business Objects* para hacer análisis multidimensional; este análisis le ofrece a los usuarios diferentes perspectivas de los datos mediante el uso de "dimensiones" que garantizan una técnica de análisis más poderosa.

Sólo para los analistas denominados trabajadores de conocimientos, no hubo una definición de alguna herramienta en especifico sin embargo se ofreció el acceso a la información con consultas estándar, el uso de la herramienta OLAP del punto anterior así como evaluar y comprar la herramienta adecuada para que pudieran encontrar a los mejores clientes para ciertos tipos de campañas. Teniendo la oportunidad de hacer análisis estadístico para encontrar tendencias, segmentación de clientes, etc.

Después de la revisión del diseño presentado, los usuarios dieron su visto bueno para iniciar la construcción del proceso de Data Warehouse.

6. CONSTRUCCIÓN

ETAPA ETL.

EXTRACCIÓN

Por las dificultades dentro de la infraestructura de hardware y seguridad en el Banco, no fue posible realizar procesos de extracción directamente a cada sistema operacional, se siguió la estrategia de solicitar a cada área de IT de cada sistema transaccional un programa de extracción nativo en cada plataforma para sacar los datos necesarios para alimentar el proceso DWH con un archivo plano con las características necesarias para que lo leyera sin problemas por el DBMS de Informix.

Los obstáculos a vencer fueron dos:

1) Buscar la prioridad para que cada área de IT atendiera cada solicitud de datos que se realizó. Para la solución de este problema fue importante contar con el patrocinio de la Dirección.
2) Dado que no se quería solicitar varios extracts por omisión de un dato, se solicitaron archivos con un detalle muy amplio lo que provocó que el volumen de información se incrementará considerablemente. Esto impactó en las transferencias, ya que estas se hacían a través de FTP por la red del Banco lo que era muy lento. Este problema se llego a controlar con un sofware que se adquirió llamado *direct connect* que ayudó a optimizar las transferencias ya que además de que la velocidad de transferencia incremento, la información viajaba encriptada y generaba códigos de error sino no transmitía al menos un byte esto ayudaba al control del proceso.

Se definió que los archivos que se entregarían bebían cumplir con las siguientes características:

a) INTEGRACIÓN: Un dato sin integración no puede llegar al ambiente Data Warehouse, ya que no puede ser usado en una vista corporativa de dato.

Afortunadamente el Banco cuenta con un sistema INTEGRADOR ("CIS" Customer Information System) que ayudó mucho, evitando la creación de procesos para identificar si el cliente del sistema Bursátil era el mismo que el cliente del Sistema de Créditos, ya que CIS se aseguraba de tener una vista unificada del cliente; por tal motivo la integración se basaría en el sistema CIS entre los campos más importantes :

- ✓ NÚMERO DE CUENTE
- ✓ NÚMERO DE CLIENTE
- ✓ CLAVE DE MONEDAS
- ✓ CLAVE DE ESTATUS
- ✓ CLAVE DE PRODUCTO
- ✓ CLAVE DE SUBPRODUCTO
- ✓ CLAVE DE SUCURSAL
- ✓ CLAVE DE PROMOTOR

Además se cuidarían los formatos de fechas:

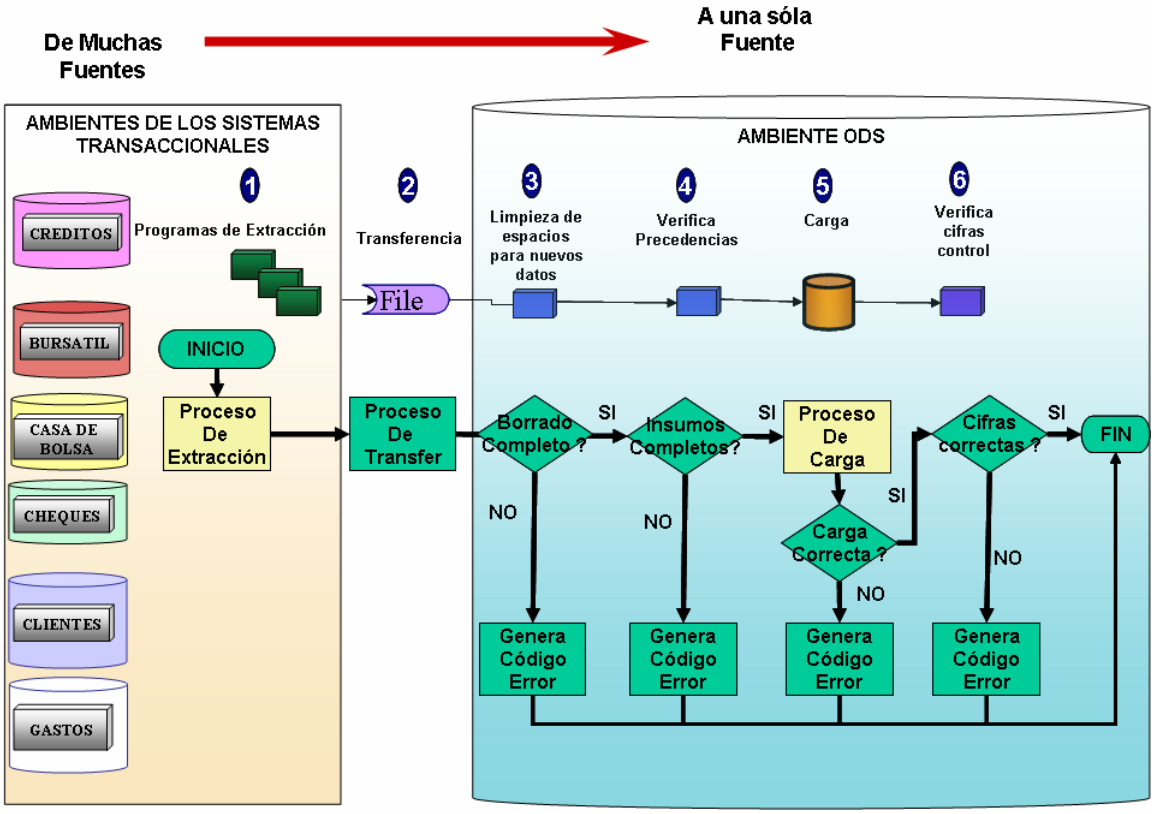
- ✓ FECHAS : Las fechas deben respetar el formato AAAMMDD
- ✓ CLAVE DE MES : Deben respetar el formato AAAAMM

b) CALIDAD. Se solicitó que todos los archivos (files) entregados incluyeran cifras control: número de registros y totales por datos que fueran importes.

c) ENTREGA. Se realizaron convenios para que la entrega del archivo fuera en tiempo y forma, es decir, que se garantizara la frecuencia de entrega en día del mes ó semana y hora, así como, respetar el layout en orden de campos y tipos.

La figura 6.1 muestra el proceso de extracción completo. El proceso de extracción realiza una replica de datos, los datos seleccionados se colocan de muchas fuentes y ambientes a un solo ambiente.

Figura 6.1 PROCESO DE EXTRACCIÓN



1.- PROGRAMAS DE EXTRACCIÓN. El proceso inicia con la selección de datos necesaria para cubrir las necesidades del modelo definido, esto se realiza a través de programas de extracción que selecciona datos dentro de los sistemas transaccionales y como resultado dejan archivos planos. La generación de estos archivos se programa de tal forma que se tiene hora y día exacto de cuando se contará con el archivo.

2.- TRANSFERENCIA. El proceso de transferencia envía el archivo al ambiente DWH salvaguardando su integridad.

3.-LIMPIEZA DE ESPACIOS PARA NUEVOS DATOS. Esta etapa borra la información más antigua, de esta forma de deja espacio para los datos más actuales.

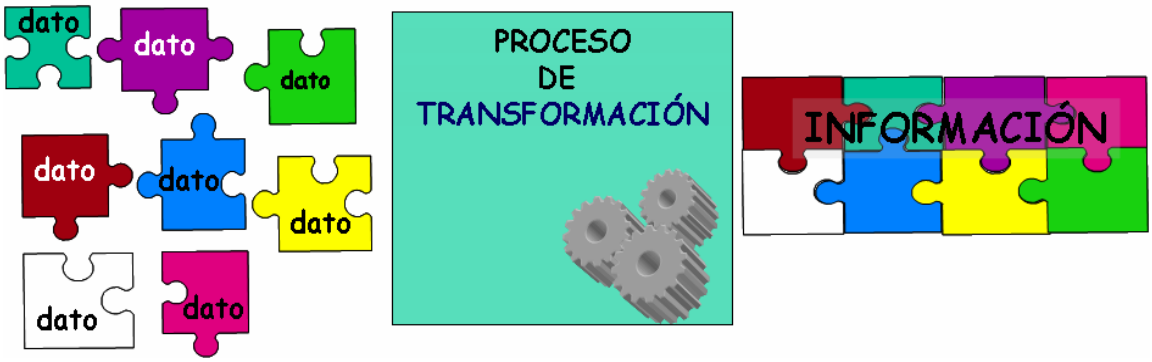
4.-VERIFICA PRECEDENCIAS. El proceso de carga simple verifica que todos las precedencias (insumos) necesarios para almacenar los nuevos datos se encuentren disponibles; estos insumos pueden ser uno o varios archivos ó tablas (éstas tablas por lo regular son catálogos).

5.-CARGA. El proceso carga simple realiza el almacenamiento en la base de datos con la sentencia INSERT de SQL. Los procesos de carga simple se programarían en Shell's ó e-sqlc (Para ver mayor detalle del tema e-sqlc ver apéndice 4 Nociones básicas de SQL embebido en C)

TRANSFORMACIÓN

El proceso de transformación convertirá los datos crudos en la información de rentabilidad que se requería.

Figura 6.2 PROCESO DE TRANSFORMACIÓN

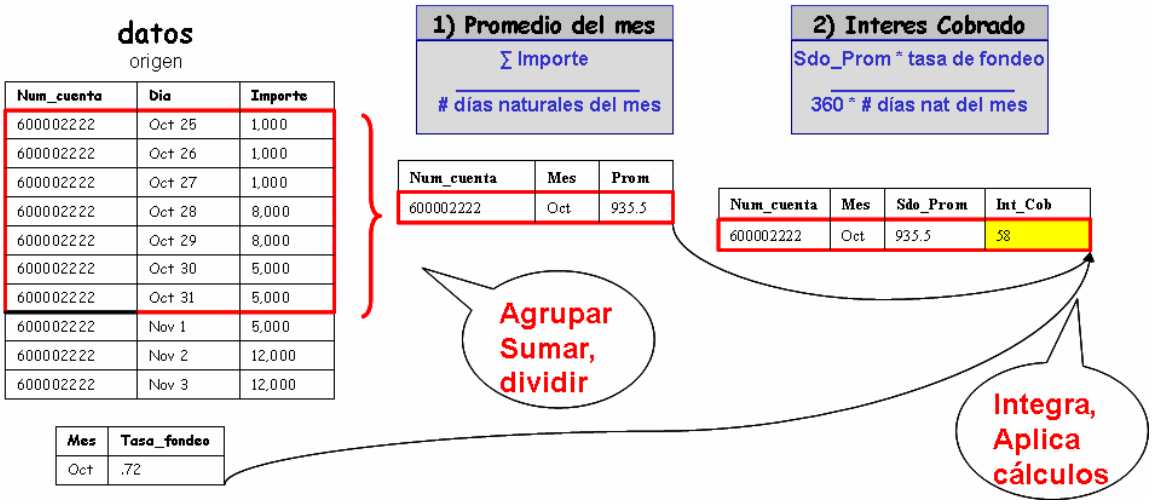


Básicamente las operaciones que se llevan a cabo en esta etapa son:

a) Integrar información de dos o más tablas a través de Join's
b) Agregar o sumarizar información.
c) Crear nuevos campos a través de cálculos empleando sumas, divisiones, multiplicaciones, operaciones con fechas, etc.
d) Filtros, elegir un universo de información, excluyendo información que no cumpla con el análisis requerido.

La siguiente figura muestra un ejemplo de proceso de transformación.

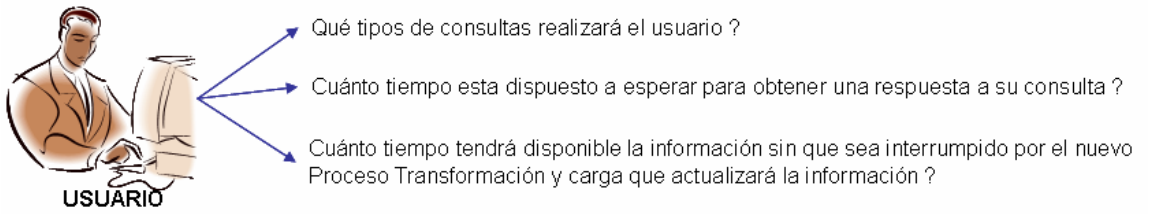
Figura 6.3 EJEMPLO : PROCESO DE TRANSFORMACIÓN PARA OBTENER EL INTERÉS COBRADO DE UNA CUENTA DE CHEQUES



CARGA

El proceso de Carga **implica más** que sólo un INSERT, esta relacionado con el rendimiento esperado por el usuario, la disponibilidad y el acceso que cubra sus especificaciones y expectativas.

Figura 6.4 PARA REALIZAR UN PROCESO DE CARGA SE DEBE SABER QUE CONSULTAS REALIZARÁ EL USUARIO

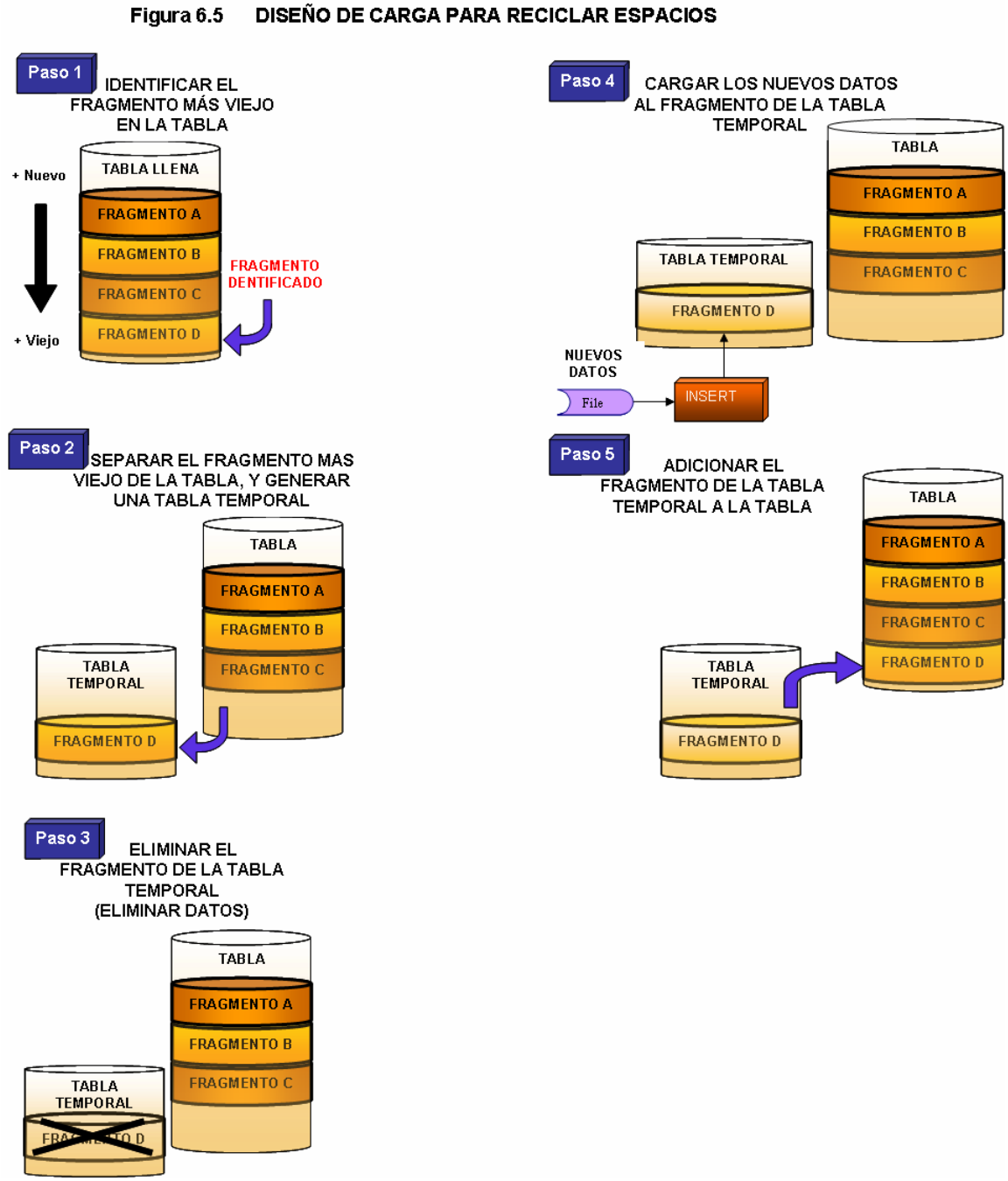


Un proceso de carga considera:

- **Tipos de Acceso** : Búsquedas por un valor concreto ó por valores en un rango especificado
- **•Tiempo de Acceso**: Tiempo que tarda en buscar un determinado elemento o conjunto de elementos **Tiempo de inserción y/o borrado**: El tiempo empleado en insertar/borrar un nuevo elemento, incluye el tiempo para empleado en buscar el lugar apropiado donde insertar el nuevo elemento ó buscar el elemento a borrar y por último el tiempo en actualizar la estructura del índice.
- **Espacio adicional requerido**. El espacio adicional ocupado por la estructura del índice

El siguiente es el diseño completo de carga en un ambiente DWH : Dependiendo de la historia que se requiriera y el rendimiento solicitado, se definió que las cargas deberían seguir un almacenamiento que recicla los espacios para poder almacenar los datos mas actuales, en este tema la fragmentación se vuelve muy relevante (para ver más detalle del tema de fragmentación ver apéndice 3 Data warehouse y esquemas de fragmentación) ya que se requiere controlar donde se almacenan los datos para reutilizar los espacios. El proceso de carga sigue los pasos que se muestran en la figura 6.5.

Figura 6.5 DISEÑO DE CARGA PARA RECICLAR ESPACIOS



GENERACIÓN DE CUBO, PUBLICACIÓN WEB Y ANÁLISIS

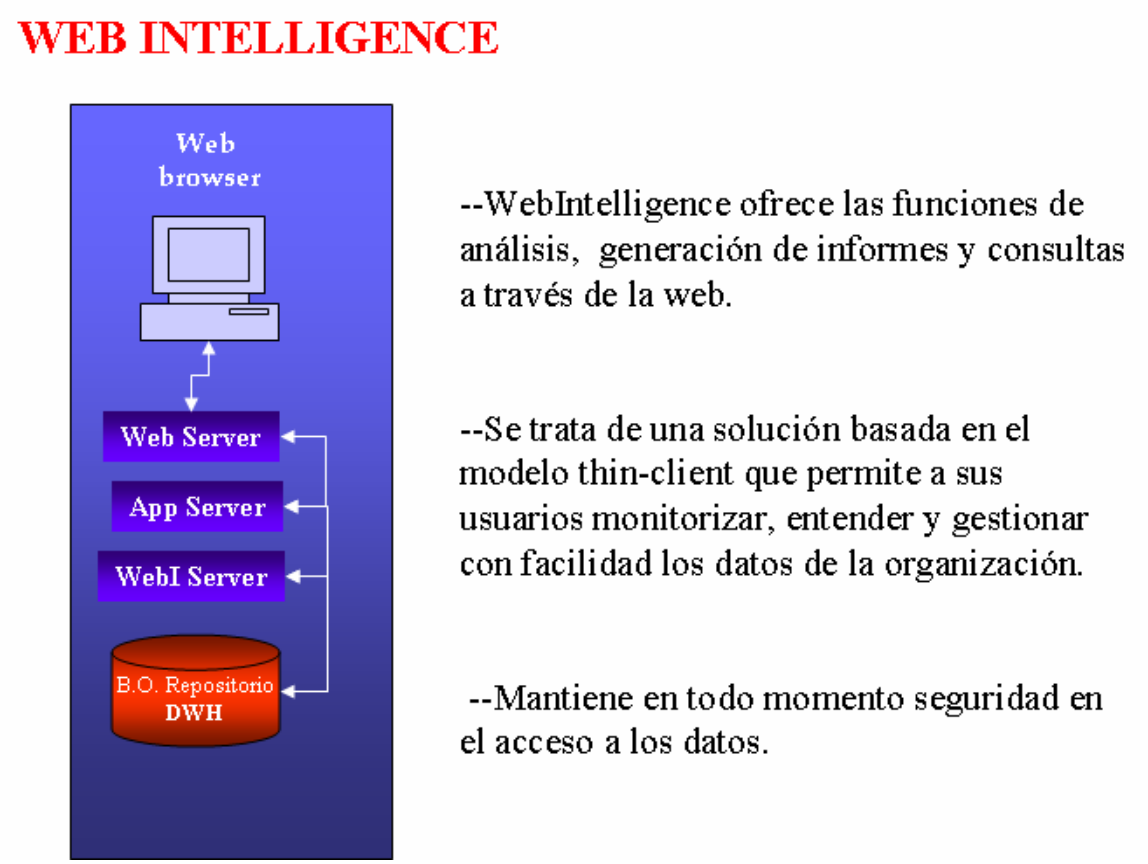
Aunque el presente trabajo no esta enfocado al delivery de los datos, será útil una breve descripción de las herramientas y la publicación web que se implementaron para entender el proceso completo que se siguió en el proyecto.

Una vez concluida la construcción de Base de Datos y la programación de extracción, carga y transformación a cargo del equipo de Modelo de Datos, el grupo de trabajo de Desarrollo Web y herramientas, inicio la ambientación del hardware en una arquitectura de tres capas. Se instalo el software de la herramienta BusinessObjects y el servidor WEB NAS (Netscape Aplication Server) que era un producto propiedad de Netscape.

HERRAMIENTA OLAP

La siguiente figura muestra la arquitectura de la herramienta OLAP.

Figura 6.6 arquitectura de 3 capas de la herramienta OLAP



Algunas de las funcionalidades de la Herramienta OLAP son:

- Es una herramienta de inteligencia de negocios
- Totalmente gráfica
- Fácil de usar

- La generación de informes o "Reporting" consiste en acceder a los datos, formatearlos y presentarlos como información de negocio tanto dentro como fuera de la Organización.
- Proporciona los elementos de información del Negocio más solicitados de una manera confiable y segura, bien a través de la web o integrándolos en aplicaciones corporativas.
- Los usuarios pueden navegar por los datos y analizarlos para conocer las causas de los mismos e identificar tendencias.
- Pueden crear sus propias consultas desde cero sin necesidad de saber SQL o conocer los complejos esquemas de base de datos.
- Pueden añadir cálculos fácilmente para realizar un análisis del negocio de manera profunda. Además, pueden compartir esa información con otras personas.
- Traducción de términos técnicos a términos de negocio "creación de universos"
- Habilita el acceso a las fuentes de información.
- Explota el poder de Data Warehouse

Publicación WEB

El equipo de trabajo de desarrollo web y herramientas diseñó y desarrolló el aplicativo llamado SIRUN (Sistema Integral de Rentabilidad por Unidad de Negocio), que permitiría la publicación de información a la máxima granularidad del estado de resultados, el principal reto sería poder llegar a todas las sucursales y ejecutivos del banco, a través de las diferentes plataformas y presentar el máximo detalle de los datos, implicando un trafico considerable por la red debido a la cantidad de información que se transfería.

La primer versión de SIRUN se realizo utilizando como servidor de WEB NAS (Netscape Aplication Server) que era un producto propiedad de Netscape, como lenguaje de programación se utilizo LIVEWIRE lenguaje propio de Netscape, como base de datos se utilizo Informix XPS, la razón principal de tomar la decisión de utilizar esta tecnología se debió a que la institución tenía como normativa desarrollar bajo este estándar y plataforma.

El resultado de este desarrollo fue mostrar los datos del modelo de rentabilidad en pantallas tipo reporte que mostraban el estado de resultado a todos lo niveles de la estructura, tenía la funcionalidad de hacer drill down[13] (bajar a un detalla mayor de información) y drill up[14] (subir a un detalle más resumido de información) sobre varias métricas. Este desarrollo implico una nueva solicitud al equipo de trabajo de Modelo de Datos, el cual consistió en realizar procesos de agregación de datos. La agregación de datos consiste en armar tablas agrupando datos a diferentes niveles, en este caso, las métricas se sumarizaron y agruparon a los siguientes niveles: Sucursal, Coordinación, Zona, Dir. Ejecutiva y Dir. Gral. Lo cual facilito el acceso a la aplicación WEB reduciendo el número de consultas (I/O) así como una recuperación de datos más optima al identificar el universo de datos que responde la consulta. La estrella de datos que se construyo ayudo mucho en este tema de "agregación[15]".

[13] **Drill Down**: Exponer progresivamente más detalle (dentro de un reporte o consulta), mediante selecciones de elementos sucesivamente.

[14] **Drill Up**, es el efecto contrario a drill down. Significa ver menos nivel de detalle, sobre la jerarquía significa generalizar o sumarizar, es decir, subir en el árbol jerárquico

[15] Agregación: Actividad de combinar datos desde múltiples tablas para formar una unidad de información más compleja, necesitada frecuentemente para responder a consultas en forma más rápida y fácil.

SIRUN dio un gran cambio de ser desarrollado en programación estructura a Programación Orientada a Objetos utilizando JAVA.

Debido a que el banco no estaba preparado tecnológicamente para los desarrollos en JAVA hubo diversos obstáculos en el camino, desde que no existía la infraestructura para la instalación del software hasta no poder contar con un ambiente de Quality Ansurance que validara los desarrollos, el trabajo fue arduo ya que a demás de lidiar con el nuevo lenguaje se tenía que cambiar la manera de programación del Banco, finalmente SIRUN versión 2 se publico en Febrero de 2000 en lenguaje POO siendo la primer aplicación del Banco en esta plataforma.

Algunas de las vistas WEB.

Figura 6.7 Ejemplo 1 vista WEB

Figura 6.8 Ejemplo 2 vista WEB

ESTADO DE RESULTADOS (CON PROVISIONES CON BANCO DEL ATLANTICO)

Sucursal: 980 CONSTITUCION

Segmento: TODOS

Concepto	I 200305	II 200304	III 200212	IV 200205	VARIACIONES I - II	 I - III	 I - IV
+ Marg.Prod.Capt.	$461,604	$538,431	$574,283	$538,431	$-76,827	$-112,679	$-76,827
+ Marg.Prod.Col.	$862,965	$825,263	$218,517	$825,263	$37,702	$644,448	$37,702
- Marg.Improd.Col.	$-9,904	$-10,757	$-6,234	$-10,757	$853	$-3,670	$853
- Marg.Imp.Inmov.	$4,923	$6,381	$7,396	$6,381	$-1,458	$-2,473	$-1,458
= Marg.Fin.Tot.	$1,309,742	$1,346,556	$779,170	$1,346,556	$-36,814	$530,572	$-36,814
+ Comi.Capt.	$111,685	$125,624	$159,835	$125,624	$-13,939	$-48,150	$-13,939
+ Comi.Banc.Elec.	$1,080,856	$1,083,389	$1,105,193	$1,083,389	$-2,533	$-24,337	$-2,533
+ Comi.Serv.Esp.	$15	$2	$8	$2	$13	$7	$13
+ Comi.Misc.	$44,333	$41,274	$43,986	$41,274	$3,059	$347	$3,059
+ Comi.Col.	$300	$50	$0	$50	$250	$300	$250
+ Comi.Comp.Vent.Div.	$17,439	$9,834	$19,601	$9,834	$7,605	$-2,162	$7,605
+ Beneficio x Infraestructura	**9,293**	**19,293**	**12,345**	**22,345**	**$-10,000**	**$-3,052**	**$-13,052**
- Costo x Infraestructura	**660,193**	**540,234**	**345,678**	**540,234**	**$119,959**	**$314,515**	**$119,959**
= Marg.Ordinario	**$1,913,470**	**$2,085,788**	**$1,774,460**	**$2,088,840**	**$-172,318**	**$139,010**	**$-175,370**
- Total Sueldo	$91,521	$315,683	$134,378	$315,683	$-224,162	$-42,857	$-224,162
- Tot.Gral.Gastos	$48,403	$37,976	$67,414	$37,976	$10,427	$-19,011	$10,427
- Gastos indirectos	**$25,340**	**$32,456**	**$12,374**	**$43,650**	**$-7,116**	**$12,966**	**$-18,310**
= Utilidad Operativa	**$1,748,206**	**$1,699,673**	**$1,560,294**	**$1,691,531**	**$48,533**	**$187,912**	**$56,675**
- Reservas	**$-340,000**	**$130,000**	**$-275,000**	**$420,000**	**$-470,000**	**$-65,000**	**$-760,000**
- Bonificaciones	$0	$707	$0	$707	$-707	$0	$-707
- Quebrantos y Castigos	$942,286	$131	$0	$131	$942,155	$942,286	$942,155
= Util.Antes.Impuestos	$465,920	$1,828,835	$1,285,294	$2,110,693	$-1,362,915	$-819,374	$-1,644,773
VOLUMENES							
Sdo.Vig.Capt.Trad	$116,661,846	$109,422,910	$113,981,424	$109,422,910	$7,238,936	$2,680,422	$7,238,936
Sdo.Vig.Capt.No.Trad.Bco	$1,634	$1,618	$227,655	$1,618	$16	$-226,021	$16
Sdo.Vig.Capt.No.Trad.CB	$0	$0	$0	$0	$0	$0	$0
Sdo.Vig.Agencias	$179,277	$124,307	$124,592	$124,307	$54,970	$54,685	$54,970
Sdo.Total.Capt.	$116,842,758	$109,548,835	$114,333,671	$109,548,835	$7,293,923	$2,509,087	$7,293,923
Sdo.Vig.Col	$328,249,332	$327,351,219	$328,040,079	$327,351,219	$898,113	$209,253	$898,113
Sdo.Ven.Col	$305,778	$1,290,187	$1,199,013	$1,290,187	$-984,409	$-893,235	$-984,409
Sdo.Vig.Col.Age	$0	$0	$0	$0	$0	$0	$0
Sdo.Total.Col.	$328,555,110	$328,641,406	$329,239,092	$328,641,406	$-86,296	$-683,982	$-86,296
Ind.Cart.Ven %	0.09	0.39	0.36	0.39	$-0	$-0	$-0
Sdo.Improd.Deud.	$4,395	$5,330	$9,150	$5,330	$-935	$-4,755	$-935
Sdo.Improd.Caja	$1,058,735	$1,059,659	$1,277,462	$1,059,659	$-924	$-218,727	$-924
Ctas.Capt.Trad	2,447	2,410	2,278	2,410	$37	$169	$37
Ctas.Capt.No.Trad.Bco	3	3	3	3	$0	$0	$0
Ctas.Capt.No.Trad.CB	0	0	0	0	$0	$0	$0
Ctas.Capt.Trad.Agencias	1	1	2	1	$0	$-1	$0
Ctas.Col.Vigente	348	397	910	397	$-49	$-562	$-49
Ctas.Col.Vencida	98	151	70	151	$-53	$28	$-53
Ctas.Col.Agen.Vig	0	0	3	0	$0	$-3	$0
No. Lideres	1	1	1	1	$0	$0	$0
No.Promotores W	2	2	2	2	$0	$0	$0
No.Promotores P	0	0	0	0	$0	$0	$0
No.Promotores A	0	0	0	0	$0	$0	$0
No.Promotores N	0	0	0	0	$0	$0	$0
No.Promotores E	0	0	0	0	$0	$0	$0
No.Promotores C	0	0	0	0	$0	$0	$0
No.Promotores G	0	0	0	0	$0	$0	$0
No.Promotores D	0	0	0	0	$0	$0	$0
No.Ejecutivos de Servicio	4	4	6	4	$0	$-2	$0

7. USO Y MONITOREO

USO

Después de pasar por una etapa de pruebas tanto del proceso ETL como de la funcionalidad de la herramienta OLAP y aplicación WEB se proporcionaron acceso a los usuarios.

Se ofreció entonces la siguiente solución a varios problemas:

- ✓ Ya no tenían que solicitar información a diferentes áreas y esperar tiempos largos para que este lista una programación de extracción de datos, los usuarios serían autosuficientes para obtener la información que requería, esto dependería de sus habilidades de los accesos disponibles.
- ✓ Se trato de proveer una solo versión de la realidad, al estandarizar la información y proporcionar las reglas y transparentar el proceso.
- ✓ Centralización de la información en un solo lugar.

No todos los accesos al usuario fueron sobre la estrella de datos que se diseño y se construyó, toda la información disponible en el almacén de datos (base de datos relacional y unimiltidimensional) fue valiosa para el usuario, enseguida se describen los accesos disponibles a los usuarios:

Consultas ad hoc para elaborar análisis de datos, para los *analistas denominados trabajadores del conocimiento* (aproximadamente de 25 a 40), se proporcionaron usuarios para acceso al equipo y la Base de Datos a través de telnet y dbaccess[16], es decir, accesaban a la información con sentencias SQL. Estos usuarios deberían conocer los datos y tener habilidades y conocimientos en SQL (Structured Query Language) de Informix para poder obtener su información, procesarla y usarla.

A través de una interfase de BusinessObjects, se dio acceso a unos cuantos usuarios (aproximadamente 15 *analistas de datos y usuarios avanzados*). Se promovió un mecanismo para que el usuario viera los datos a un alto nivel y que entonces obtengan con ello la solución a preguntas específicas. Esta herramienta se conectaba al modelo multidimencional, y a la base de datos relacional, le permitía al usuario armar reportes dinámicos haciendo drill down/drill up yendo de datos resumidos al detalle incluso de cuenta y cliente y viceversa.

---

[16] **Dbaccess** es una herramienta proporcionada por el SGBD de Informix que facilita el acceso y administración a una Base de Datos Informix, sin necesidad de desarrollar una aplicación.
Con dbaccess se puede:
Crear y borrar Bases de Datos (Para esto debes tener los permisos adecuados)
Crear, modificar y borrar tablas (Para esto debes tener los permisos adecuados)
Cargar Datos desde archivos del sistema Operativo
Introducir, modificar, consultar y extraer datos de las tablas de la Base de Datos
Crear y eliminar Índices
Obtener información acerca de las bases de Datos y sus tablas

A los *usuarios de negocio* se les capacito en la **aplicación WEB** para que dominaran la navegación y pudieran consultar sus datos en cualquier sucursal del Banco a nivel Nacional.

USO DEL MODELO DE RENTABILIDAD.

El Modelo de Rentabilidad permitió al Negocio tener el conocimiento de las unidades de negocio que aportaban Valor (por vistas de sucursal, producto y promotor). Debido al diseño de la Solución fue posible realizar análisis ad hoc a todos los niveles, por fin la información llegaba a todos los colaboradores dentro de la institución, la gestión basada en la información flexible permitía tomar decisiones más asertivas trayendo beneficios como:

- ✓ Asignación adecuada de Recursos, se tomaban decisiones de cerrar sucursales poco rentables y asignar más recursos a aquellas de tenían potencial para crecer o eran buenas.
- ✓ Cobro de comisiones de servicios a clientes que no dejan rentabilidad adecuada pero hacen uso de la infraestructura de la institución.
- ✓ El modelo de Rentabilidad se convirtió en una plataforma para elaborar los presupuestos de ingresos y egresos.
- ✓ Ayudo afinar el Modelo de incentivos a las fuerzas de ventas (ejecutivos), se pagan bonos (incentivos variables) además de su sueldo fijo, una variable importante fue la rentabilidad de la cartera del promotor y la sucursal.
- ✓ Como el modelo consideraba el nivel cliente, se tenía el conocimiento del valor de cada cliente, con esta información fue posible despegar las iniciativas de CRM[17] en el Banco.

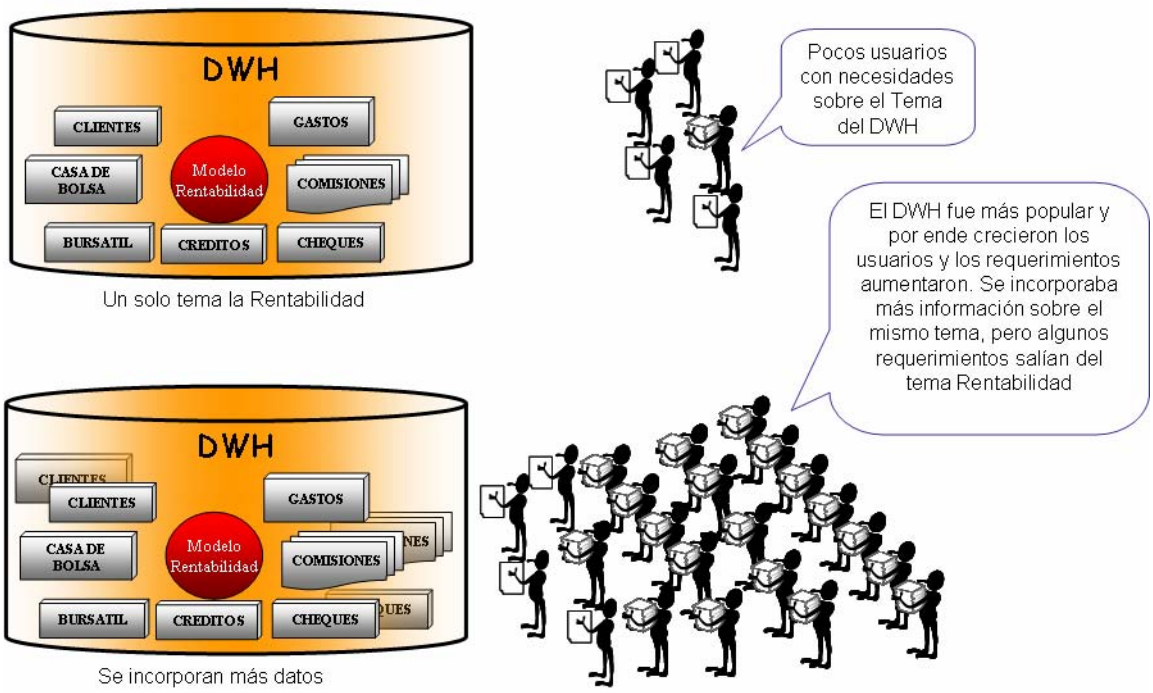
El uso por todos los medios del modelo de rentabilidad trajo consigo una sería de necesidades nuevas, además la dinámica del negocio provoco el mantenimiento temprano a las reglas de negocio en el Modelo para mantenerlo al día.

USO DE LA INFORMACIÓN EN GENERAL.

Aunque el tema del Data Warehouse fue la Rentabilidad, se volvió muy popular entre todos los colegas dentro del Banco; la información centralizada en un solo lugar y estandarizada y sobre todo el nivel de granularidad de la información provoco que muy pronto todo el personal que requería información se acercara a solicitar acceso y obviamente solicitaba la integrar nueva información para complementar sus reportes.

[17] CRM Customer Relationship Management: La administración de las relaciones con el cliente, es la DISCIPLINA para RETENER clientes y volverlos más rentables, mediante el APRENDIZAJE de algo nuevo en CADA CONTACTO, a través de campañas adecuadas.

Figura 7.1 CRECIMIENTO DE USUARIOS QUE EMPLEAN LA INFRASTUCTURA DE DWH



Con el crecimiento de usuarios a quien atender, fue necesario formalizar el equipo de trabajo que atendería las necesidades de información. Se creo una gerencia de Data Warehouse (esta gerencia se formo básicamente con el grupo de Modelado de datos) que tenia como responsabilidad principal actuar como interfaz entre el equipo técnico del Data warehouse y sus usuarios finales. Este grupo trabaja en coordinación con los usuarios finales (negocio) para identificar y documentar las necesidades de información y, basadas en estos requerimientos discutir y revisar que la solución propuesta cumpla con las necesidades del Negocio.

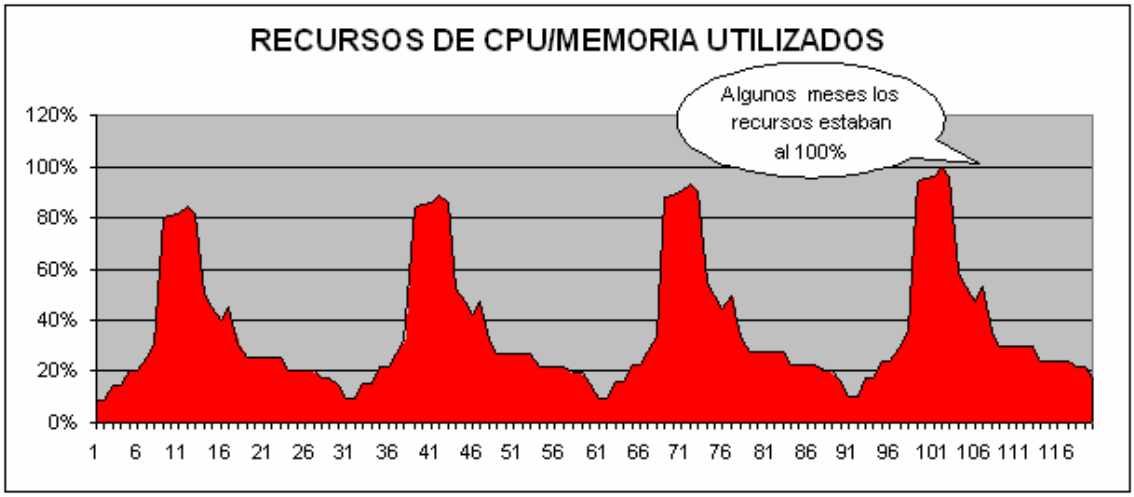
MONITOREO

La gerencia implemento una sería de mediciones que le ayudaran a identificar y prevenir cualquier problema con el proceso.

Al equipo.

El monitoreo que se realizó al equipo traía algunas preocupaciones para algunos meses (sobre todo Julio / Diciembre) se presentaban problemas de recursos. En la siguiente gráfica se muestra la utilización de hardware, misma que muestra un comportamiento interesante característico de una plataforma DWH, ya que los días donde se realizan más procesos y consultas por parte del usuario son precisamente cuando corre el proceso mensual, ya que gran volumen de información es procesada y actualizada para el usuario. Se nota que va bajando tanto el procesamiento de datos como las consultas de los usuarios y pareciera que el equipo esta siendo subutilizado pero esto no es así, es normal que en plataformas DSS suceda este comportamiento.

Figura 7.1 Utilización del Hardware en la plataforma DWH



Al proceso.

Cada mes se generaba medidores de truenes productivos (se llamaba truene productivo aquel proceso que por alguna razón no llegaba a buen fin ó bien el usuario reportaba como datos no confiables o con mala calidad), se daba seguimiento y se ejecutaban acciones sobre los más recurrentes y los que se presentaban con mayor frecuencia. La siguiente tabla muestra los errores más comunes que se presentaron.

Descripción del problema	%
Problemas con archivo fuente (atribuible al sistema operacional fuente)	13%
Problemas con espacios de la Base de Datos (atribuible al crecimiento en volumen de algún archivo)	2%
Problemas programa de carga (atribuible al cambio de alguna regla de negocio)	5%
Problemas de transferencias (Muy lentas y se interrumpen)	80%

Tabla 7.1 Principales problemas en una plataforma DWH

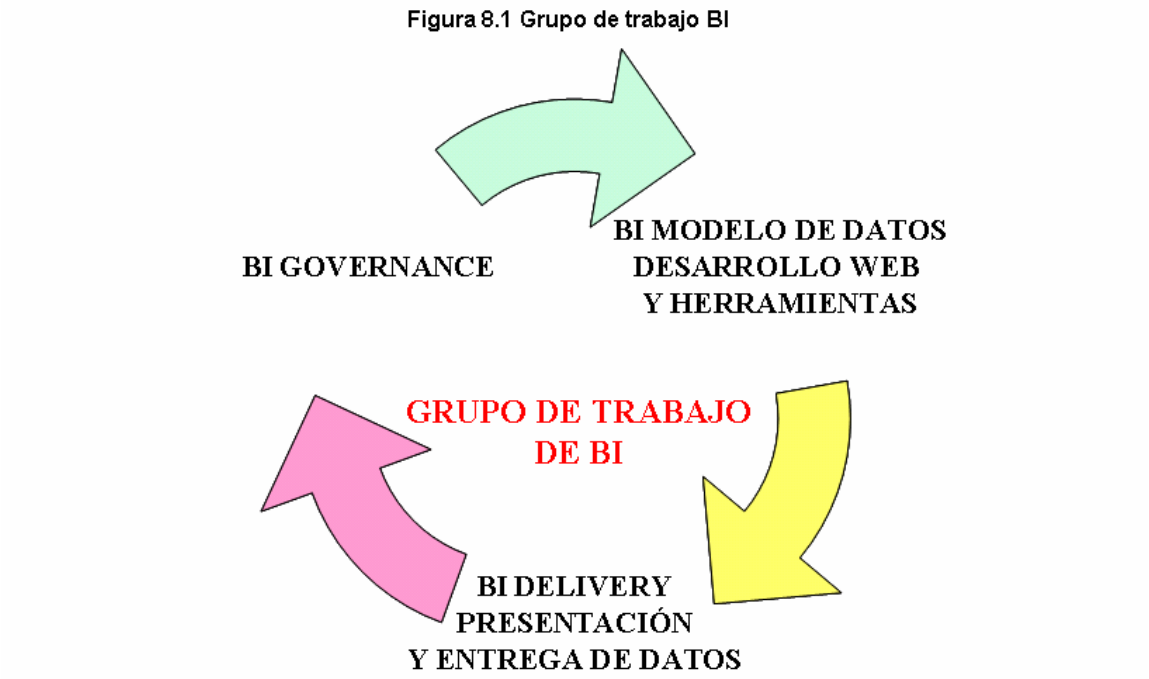
La nueva gerencia tuvo que solicitar al equipo de soporte de tecnología DWH una solución para un problema que ***además de recurrente, bajo el rendimiento del servicio significativamente y estaba provocando un problema de credibilidad muy grande ante los usuarios.*** El apéndice 4 muestra El Problema de transferencias, solución a un problema crítico.

8.EVOLUCIÓN

En Agosto del 2004 se propuso la evolución de una plataforma Data Warehouse (con el tema de rentabilidad) a una plataforma Business Intelligence. Este capitulo describe a detalle la propuesta realizada.

Como primer punto se toco el tema de roles para conformar un equipo más sólido de trabajo. El equipo se conformaría de la siguiente manera:

Figura 8.1 Grupo de trabajo BI



BI GOVERNANCE

Es un Comité Directivo. El objetivo del comité es supervisar y gobernar el proceso BI, esta involucrado en muchas cosas, desde la aprobación del presupuesto hasta la definición de nuevos requerimientos, examina los problemas del proceso Data Warehouse para asignar recursos, prioridades, decidir que solicitudes se deben atender de acuerdo al beneficio del Negocio, etc. Este comité esta integrado por el patrocinador(es) ejecutivos, los administradores del proyecto y un grupo selecto de usuarios finales.

Se establecieron dos tipos de reuniones para el BI Governance, una periódica cada semana donde sólo participan los integrantes del comité y otra con periodicidad mensual (o sólo cuando sea requerida) donde se enlaza con los comités de Tecnología y de rumbo del Banco.

BI MODELO DE DATOS, DESARROLLO WEB Y HERRAMIENTAS

En realidad son dos equipos, pero coexisten con el mismo fin: distribuir información confiable que pueda ser accesada oportunamente (En estos equipos de trabajo es donde más ampliamente he trabajado).

MODELO DE DATOS. Entre sus responsabilidades más importantes están:

1. Administración de la información (datos): Es función primordial del área decidir con exactitud cuál es la información que debe mantenerse en las Bases de Datos, es decir, identificar las entidades de interés común de los usuarios, realizando un estudio previo de necesidades de información de los clientes, para elaborar un Diseño lógico o conceptual de la Base de Datos, ya sea para actualizar, borrar ó ingresar información.
2. Establecimiento de políticas para mantener la integridad de la información, Estableciendo las políticas para mantener, manejar los datos y mantener la seguridad de la información.
3. Modelado de Datos (ya sea relacional o multidimensional), diseñar la Base de datos para garantizar que el usuario tendrá el mejor acceso a la información.
4. Trabajar estrechamente con el Analista de Negocio para entender y documentar las necesidades de los usuarios, para identificar y analizar los datos de apoyo a las decisiones, esto incluye Actuar como el enlace entre el Equipo desarrollo WEB y herramientas y los usuarios finales del DW.
5. Entender los requisitos de información por lo que se refiere a las unidades de negocio específicas. Traduciendo y descomponiendo esos requisitos para que cada uno de los participantes tanto de Negocio y técnico lo entienda cada quien en su contexto. Asegurar que las necesidades de negocio se incorporen a las soluciones técnicas definidas.
6. Diseñó del proceso de Adquisición de Datos. Identificar fuentes apropiadas de datos para alimentar el Data Warehouse, Identificar oportunamente los obstáculos involucrados para obtener los datos necesitados por el negocio
7. Modelado de datos lógico para un Data Warehouse (con el conocimiento de las diferencias entre un ambiente Data Warehouse y un ambiente operacional tradicional). Esta responsabilidad incluye la identificación de datos más valioso a la corporación, la integración de estos datos, y el mapeo del dato para ponerlo en correlación al modelo de los datos.
8. Participar en la comprobación del Data Warehouse. Validar que la solución del Data Warehouse satisface los requisitos de los usuarios finales, es decir, asegurar que el extracto de datos del sistema fuente y el movimiento al Data Warehouse se realicen "correctamente", y se realicen las transformaciones u otras reglas de negocio aplicadas a los datos "correctamente." Correctamente se define como "de acuerdo con los requisitos de usuarios finales identificados".
9. Documentar procesos y los datos del Data Warehouse.
10. Estimar el tamaño de la aplicación del Data Warehouse , Estimación de costos para la soluciones técnicas y evaluación y selección de hardware y software

DESARROLLO WEB Y HERRAMIENTAS. Es un equipo muy técnico tiene estrecha relación con proveedores de herramientas, dba's y programadores, entre sus responsabilidades más importantes están:

1. Mantener el modelo físico de datos del ambiente data warehouse, recomendar la mejor estructura de almacenamiento de datos del data warehouse.
2. Dentro de sus responsabilidades se encuentran aspectos tanto estratégicos como tácticos, relacionados con la implantación de la arquitectura técnica del

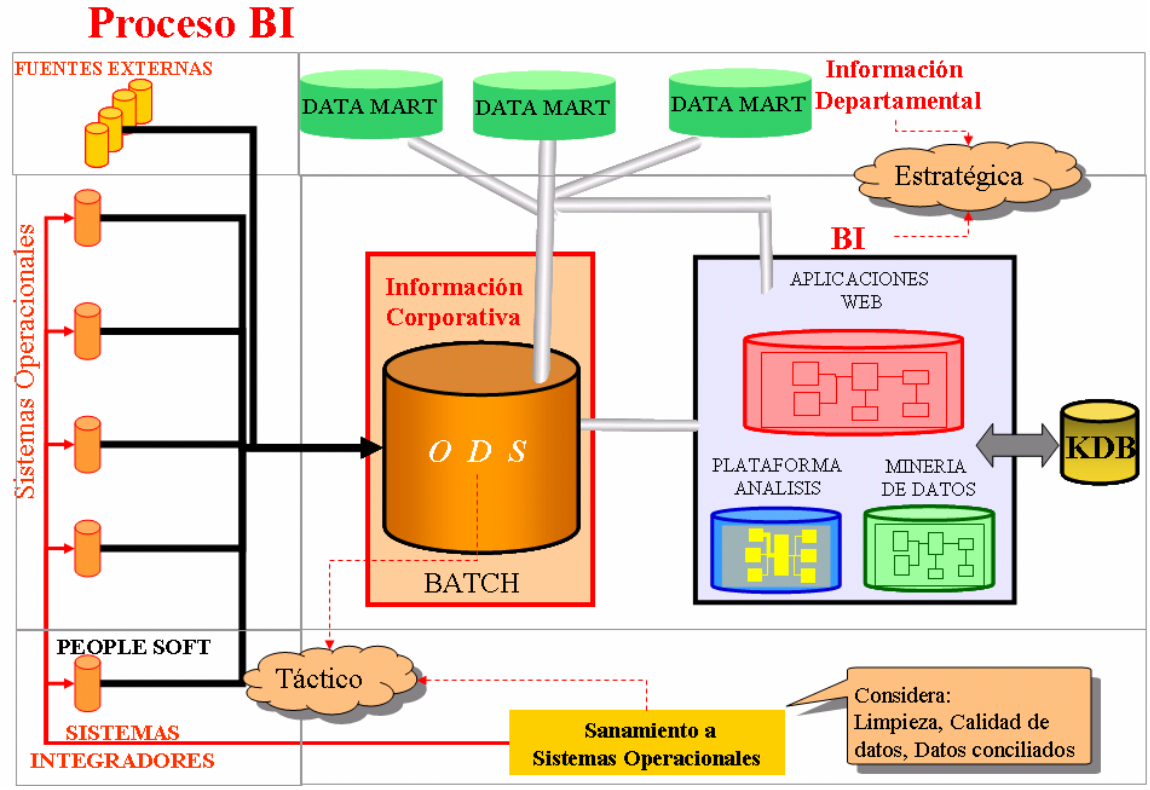
Data warehouse. Entre estas responsabilidades se encuentra la evaluación y selección de software de base de datos, herramientas de almacenamiento, performance de herramientas ETL, software de business intelligence.
3. Seleccionar y configurar estas herramientas formando diferentes capas de software de tal forma que se obtenga una arquitectura escalable, abierta y acorde a las necesidades de la organización. Esta arquitectura establece la plataforma base sobre la cual se dispondrán los datos, funciones y herramientas para que el Data warehouse ofrezca sus servicios de información.
4. Diseño de algoritmos eficientes y dinámicos que resuelvan los requerimientos de negocio. Otra responsabilidad importante es la evaluación y selección de software de automatización de procesos, scheduler tools y metodologías de programación, herramientas ETL. Configurar estas herramientas formando la biblioteca de funciones del data warehouse clasificada por capas de software (batch, calculadora, extracción, etc).de tal forma que se obtenga una arquitectura de código escalable y abierto, acorde a las necesidades de la organización. Esta arquitectura establece la plataforma de funciones que deberán utilizarse en todos los requerimientos a la plataforma Data warehouse.

BI DELIVERY PRESENTACIÓN Y ENTREGA DE DATOS

Es un grupo de expertos, quienes conocen muy bien el Negocio y pueden adelantarse a las necesidades de los usuarios, entre sus responsabilidades más importantes:
1) Diseño Final de las aplicaciones y reportes,
2) Realiza la difusión de las herramientas,
3) Elabora manuales y cursos para esta difusión,
4) Es el primer contacto con los usuario finales para resolver dudas o dar soporte, 5) Realiza análisis haciendo proyecciones,
6) Diseñando modelos predictivos y colocando en cada reporte entregado un análisis financiero y una recomendación ó conclusión de la información presentada.
7) Este grupo de trabajo es quien realmente entrega inteligencia al Negocio y están a cargo de la KDB (Base de Datos de Conocimientos) de la plataforma.

Enseguida se describirá la propuesta del proceso Business Intelligence, que se definió, la siguiente imagen muestra el proceso de BI.

Figura 8.2 Proceso BI



Las estrategias planteadas para llevar a cabo este proceso fueron las siguientes:

EN CUANTO AL HARDWARE SE REFIERE, FUERON TRES LAS ESTRATEGIAS SUGERIDAS:

Optimizar. Se refiere a una revisión de la información existente, donde se haga una evaluación de los procesos existentes y la historia requerida, con el fin de evitar tener información no requerida, duplicada o sin uso dentro de los equipos del BI. También es necesario hacer una depuración de procesos que actualmente no requiere el negocio o aportan muy poco valor y que son un factor de riesgo para el seguimiento de los entregables, además de ocupar recursos técnicos y humanos para dar seguimiento al proceso productivo.

Ordenar. Se encontró una mezcla de plataformas (ambiente operativo y analítico), se sugiere una separación de estos ambientes para evitar un uso ineficiente de los recursos actuales, como se habla en el marco teórico de este trabajo las plataformas de un ambiente OLTP y un DSS son muy diferentes por tal motivo no hay razón alguna para permitir que una arquitectura de la plataforma operativa se mezcla con la del análisis, ya que va provocar serio problemas de flexibilidad, acceso y disponibilidad de los datos.

Remplazar y hacer crecer. Muy pronto los equipos serán insuficientes para abastecer la demanda de procesamiento de información y esto será un ciclo constante por lo que se sugiere realizar presupuestos para los siguientes 5 años en la plataforma y de esta manera permanecer con equipos actualizados adecuados a las demandas del negocio sin necesidad de hacer una solicitud de presupuestos a la Dirección por cada proyecto que presente un usuario. Aquí el tema es realizar un análisis adecuado del hardaware para saber si es suficiente hacer crecer el o los equipos en disco, cpu's o

memoria o si es necesario reemplazar el o los equipos escalándolos a tecnología de punta. Para esta tarea es necesaria la colaboración estrecha de las áreas de soporte expertas de IT.

EN CUANTO AL TEMA DE HERRAMIENTAS, LAS SIGUIENTES FUERON LAS PROPUESTAS:

Antes de llegar a las propuestas sugeridas se realizo un análisis basado en el Alcance del proceso BI (de acuerdo a la demanda de requerimientos del negocio), se tomaron en cuenta los riesgos y se discutieron los criterios de éxito para cada alternativa de solución. Después de este proceso se llego a las siguientes propuestas.

Manejador de Base de Datos.

En cuanto a la herramienta DBMS la propuesta consiste en conservar las plataformas batch y operacional con IBM Informix XPS debido a los tiempos de respuesta que ofrece en las cargas masivas, el manejo de la historia y principalmente por la biblioteca de funciones y algoritmos en ESQLC, korn shell desarrollados bajo este manejador.

La plataforma de análisis web estará tanto en IBM Informix XPS (desarrollos anteriores) como en DB2 UDB (desarrollos nuevos) y en una segunda fase se haría un análisis para observar la factibilidad de migrar los desarrollos en IBM Informix XPS a DB2 UDB u Oracle.

Adicionalmente se propuso incorporar herramientas para el modelado de Datos, como **ERWIN** (herramienta CASE en modelos ER).

Minería de datos.
El ambiente de minería de datos estará bajo DB2 UDB y DB2 Intelligent miner for data. DB2 intelligent miner es una solución de minería de datos que forma parte de la suite DB2 Data Warehouse Entrerprise Edition. La familia de productos de DB2 Intelligent Miner apoya a las empresas a identificar y extraer valor a la inteligencia del negocio mediante la clasificación de datos.

Herramientas ETL.
La propuesta consiste en generar una lista de herramientas en las que se basará este proceso, donde se contemple bajo que condiciones del requerimiento se utilizará ESQLC, korn, shell, etc. Se propone incorporar la herramienta **DataStage**, con esta herramienta se espera eliminar los tiempos de programación, ya que es una herramienta visual donde sólo el diseño del proceso ETL es basado en tomar y arrastrar objetos y una vez hecho el diseño sólo se define su calendario de ejecución, además se tiene la facilidad de la administración de los Metadatos y la documentación con la misma herramienta del DataStage y construir procesos de calidad de datos con **Quality Manager** que también incluye DataStage.

Herramientas OLAP y Reporteadores.
Para el caso de motores de OLAP, la herramienta que se propone es Hyperion integration Server, ya que es una suite de herramientas gráficas de servicios de integración de datos, escalable, que minimiza el tiempo utilizado en crear, explotar y manejar aplicaciones analíticas (Como es el caso de Business Objects y COGNOS).

Hyperion Integration Server utiliza un metadata centralizado creado automáticamente que puede ser utilizado en todos los componentes de la suite; su utilidad es entre otras la reutilización de metadatas en la creación y manejo de aplicaciones analíticas cuyas

fuentes pueden ser datawarehouses, datamarts, aplicaciones transaccionales, ERP's, etc.

Hyperion Integration Server permite a los usuarios navegar desde datos sumarizados, derivados y calculados desde la herramienta (drill down), hasta el detalle de los datos almacenados en bases de datos relacionales (Drill through).

La tecnología de almacenamiento crea realmente "modelos multidimensionales de información, donde cada una de las celdas del modelo corresponde a un cruce de métricas VS dimensiones. Los conceptos de tablas y campos desaparecen con esta tecnología. Esta herramienta también es distribuida por IBM y se conoce como DB2 Olap Server, pero la funcionalidad es la misma.

Herramientas de delivery.

Se propuso elaborar un inventario de soluciones, donde se pondrán las herramientas que hayan pasado por una evaluación exhaustiva tanto de prueba de la herramientas como de selección de un proveedor para soporte y outsourcing que de seguimiento y soporte a toda la plataforma BI. Se propone iniciar con las siguientes:

La Suite de Cognos.
La Suite de Business Objects.
La Suite de Actuate.

Aplicaciones front-end y soluciones existentes en el mercado.
Para las aplicaciones de tipo front end que empleen la Intranet del Banco para su distribución, se propuso respetar las políticas actuales, las cuales son Java, utilizando WebSphere de IBM como servidor de aplicaciones y según sea el caso MQSeries. Adicionalmente se utilizará UML como la metodología de trabajo. En caso de requerirse alguna solución que exista en el mercado como aplicaciones especializadas al BI, como Balanced Scorecard, iniciativas de ABC, Planning, etc. Se propone seguir el mismo paso que en las herramientas de delivery.

EN CUANTO A LA ESTRATEGÍA DE DATA MARTS:

Con el fin de cumplir con las necesidades de los usuarios y el banco en general, el modelo de datos corporativo deberá ser ajustado a 4 funciones básicas a cumplir para empresas Financieras, estas son:

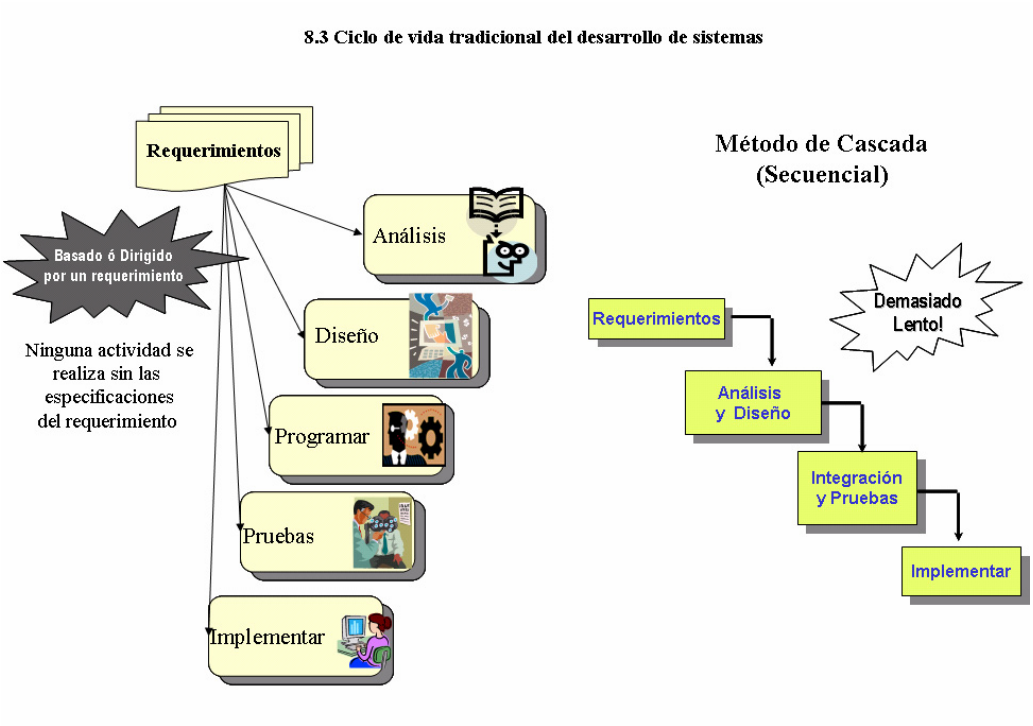
a. Fianzas y Rentabilidad (presupuestos).
b. Riesgos.
c. Relación con el Cliente y mercado (CRM).
d. Organización (Tesorería, operación y regulatorios, Medición del Negocio en sus Variables más representativas Balanced Scorecard)

Este modelo se implementará según las necesidades locales, pero el modelo local (Data Mart) deberá ser congruente en un solo requisito, la fuente para alimentar a todos los Data Marts será el repositorio corporativo (ODS).

Es importante resaltar que el Banco ya contaba con soluciones de CRM y Balanced Scorecard. Lo que se espera con la propuesta es proporcionar una plataforma más grande y sólida para que estos procesos sigan avanzando y dando resultados.

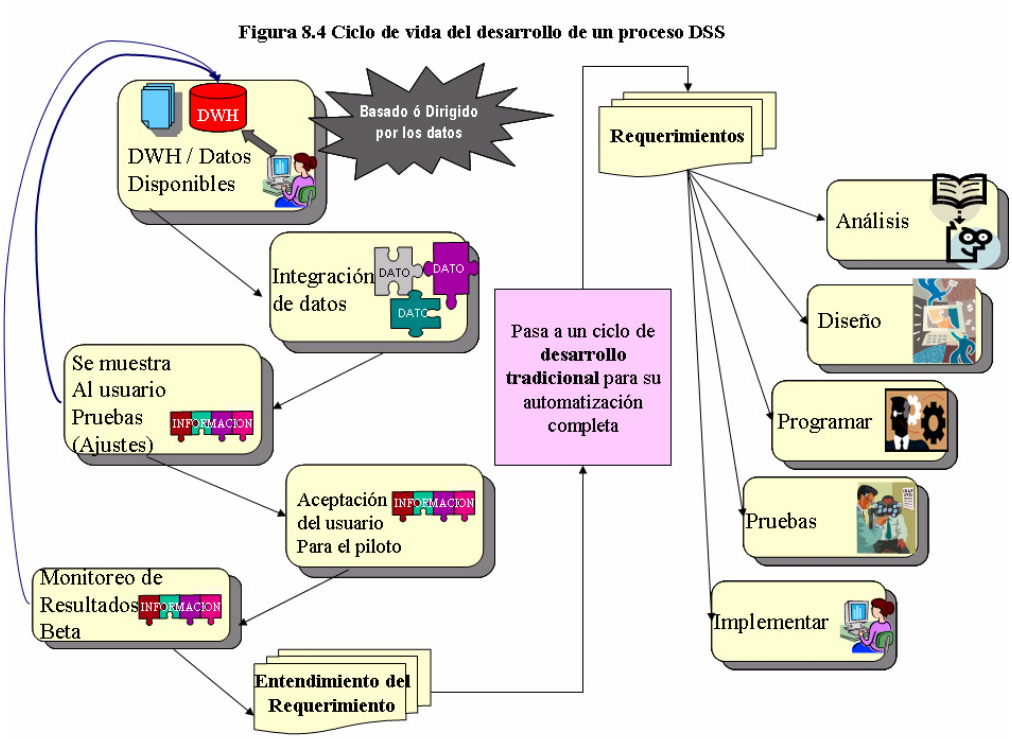
EN CUANTO A LA METODOLOGÍA DE TRABAJO:

Se propuso romper con la metodología tradicional de IT basada en un requerimiento y con un método de cascada secuencial, que resulta ser muy lento para proyectos DSS.



Cuando se trabaja con un proyecto DSS no se conoce el requerimiento con exactitud, en cambio cuando hablamos de los sistema operacionales podemos decir que el ciclo de vida tradicional del desarrollo del sistema es adecuado porque el requerimiento es muy preciso, esto se debe a que el diseño esta basado en automatizar procesos de negocio cuyas reglas de negocio son más estables a lo largo del tiempo, lo que conlleva a una mejor definición de sus requerimientos; un sistema DSS en cambio, está orientado a mejorar el proceso de toma de decisiones, el cual resulta ser un proceso muy variable a través del tiempo debido a las diversas y cambiantes situaciones en las cuales de deben analizar los datos. Es por ello que se propone seguir el ciclo de vida del desarrollo de un proceso DSS.

Figura 8.4 Ciclo de vida del desarrollo de un proceso DSS



Aunque aparenta ser complicado en realidad es un proceso simple si se cuenta con el prerrequisito de contar con los datos disponibles. No se puede llegar a un buen requerimiento de un proceso DSS si no se cargan e integran los datos y se muestran a los usuarios. La única forma de alcanzar un diseño confiable es crear un ciclo de retroalimentación temprana en el proyecto. Este ciclo de retroalimentación utiliza a los usuarios finales para evaluar los prototipos construidos (pilotos), en un proceso iterativo que continua a lo largo del proyecto. De esta manera el equipo de Modelado de datos puede vigilar la aceptación del usuario y descubrir defectos de diseño en forma temprana. Es importante mencionar que la aceptación del usuario se verá limitada si los datos que se le muestran son sólo una muestra ya que no sabrá si cubre sus requisitos, por tal motivo es imposible que el usuario acepte que el proceso es correcto sino cuenta con datos reales (esto tal ves involucre un gran volumen de datos).

Lo descrito en este capitulo es solo una propuesta de la Evolución para el DWH en el Banco Bital (ahora HSBC), fue una propuesta elaborada por el personal que estuvo desde el arranque en la construcción e implementación del DWH (con el tema de Rentabilidad), quienes llevaron al proceso DWH a ser una sólida plataforma de datos para los procesos de toma de decisión. Esto habla de 5 años de experiencia en estos procesos, y un arduo trabajo en el procesamiento de datos y sistemas de información.

Todavía no es claro cuál será la conclusión de la propuesta ya que llevará tiempo tener el patrocinio de la Dirección (para cubrir los gastos tecnológicos y el apoyo de la nueva cultura), así como poner de acuerdo a todas las áreas de Negocio, Soporte e IT en un solo rumbo. Sin embargo será muy difícil que se frene la evolución del DWH ya que es una herramienta y proceso de misión crítica para el Banco por la gran aportación que da esta tecnología la cual ha ayudado a los usuarios; que al final del día son los responsables de gestionar para obtener los resultados, mantener las ventas, incrementar los ingresos y tomar las mejores decisiones para el futuro del Banco.

9. CONCLUSIÓN

Tal ves las iniciativas planteadas en el capitulo de evolución cambien un poco en su ejecución pero dos premisas dentro de mundo de BI no debe perder en cuenta el lector.

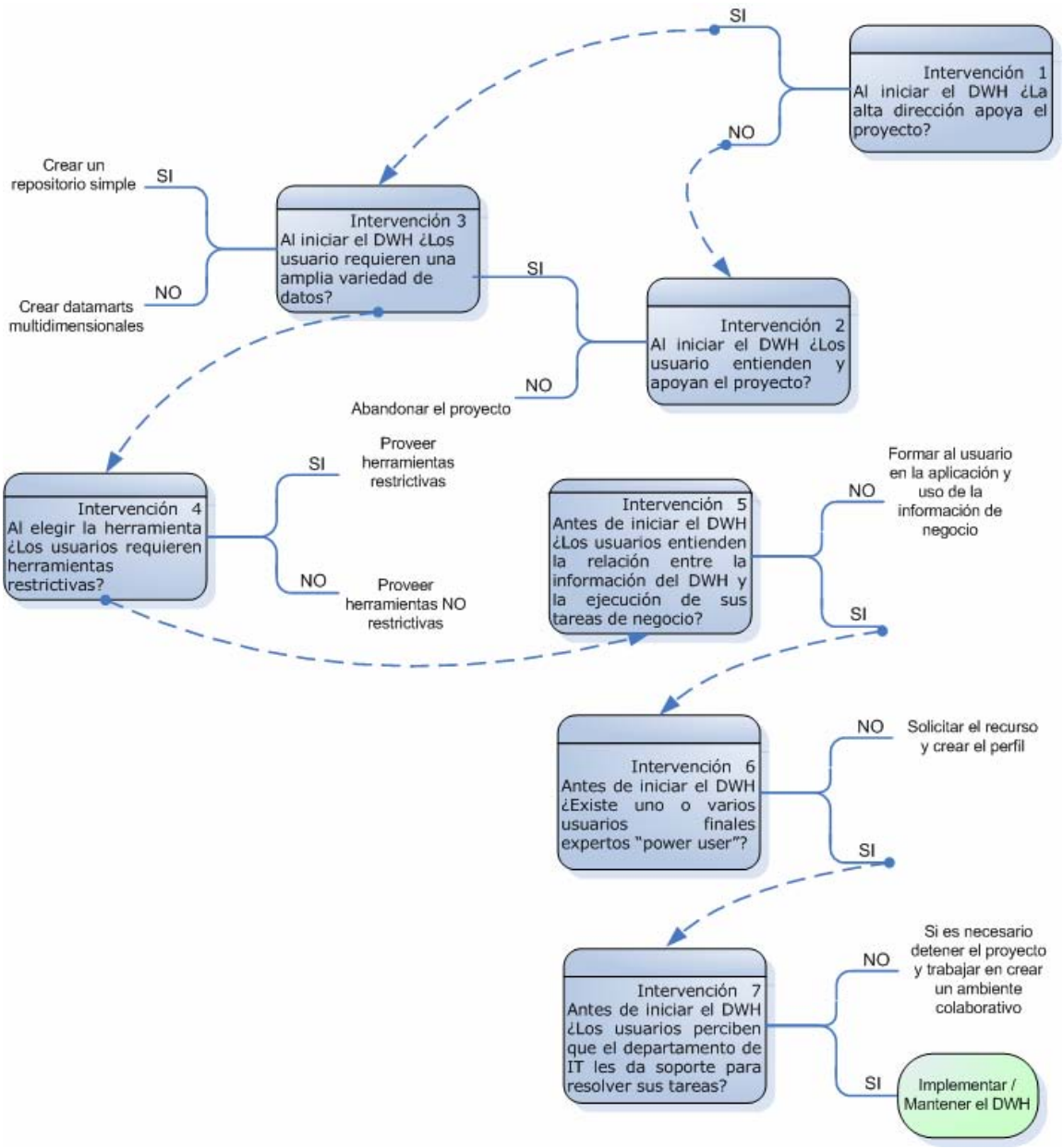
La evolución de un DWH debe hacerse en incrementos, conformado por una serie de pequeños proyectos (temas de negocios afines), con cada uno de ellos agregando un tema a la estructura del Data Warehouse como parte de una estructura planeada. Y esto solo se puede dar si el profesional informático que participe en el proyecto, tiene un conocimiento suficiente del tema de negocio que contemplará, esto es necesario ya se debe tener por lo menos una proyección de los requerimientos futuros para poder darle un cierto nivel de flexibilidad y velocidad a los modelos relacionales o multidimensionales que se diseñen. Esto se traduce en enriquecer los modelos dentro del DWH y no hacer crecer el la estructura de Data Warehouse por cada requerimiento del usuario por la falta de conocimiento de la información.

Actualmente laboro en un departamento que es usuario de la infraestructura de DWH descrita en este trabajo, pero sigo trabajando en modelos de BI seguramente al igual que a mi el termino BI para mucho es ambiguo, lo único claro es que un termino tipo paraguas bajo el cuál se cobijan varias disciplinas y herramientas como OLAP, Datawarehousing, Datamarts, Minería de Datos, Decisión Support Systems, Redes Neuronales, Sistemas Expertos, Cuadros de mando, Balanced Scorecards, Customer Relationship Management (CRM), etcétera, etcétera. Yo trabajo con Datamarts y Balanced Scorecards actualmente y el almacén de datos corporativo que es ahora el DWH es de mucha utilidad y se fortalece cada día con todas las iniciativas de BI que alimenta, para mayor comprensión del término BI es posible identificar que una herramienta es de BI si cumple con los propósitos:

a) De **proveer de información** para la gestión, el control y la medición del negocio. Para controlar y medir una herramienta no puede quedar en presentar información, también debe permitir al usuario final manipular y navegar sobre los datos para que pueda realizar el análisis adecuado.
b) Agrupar esta información y ponerla en el **tiempo adecuado** a disposición de la gestión del negocio. La ayuda a la toma de decisiones es sin duda la característica para importante de BI.
c) Poner a disposición del usuario información que le sea clara, el usuario no debe tener **problemas de semántica** al interpretar un dato, los datos que use deben estar en leguaje de negocio, es decir, datos de su total comprensión.

Como conclusión final de este trabajo y de acuerdo a la experiencia adquirida en este proyecto de construcción de un modelo de rentabilidad basado en un Data Warehouse Concuerdo completamente con el articulo "Seven Key Interventions for Data Warehouses Success", publicado en el año 2006 en Communtications of the ACM por Tim Chenoweth, Karen Corral y Haluk Demirkan, que describo enseguida y que ayudo a que el proyecto que describe este trabajo no haya fracasado.

Las 7 intervenciones para el éxito de un DWH se basan en el uso de la tecnología con el contexto social (clima laboral) y corporativo entendiendo como éxito en que el DWH se use y como fracaso el que no se uso.



Intervención 1: Al iniciar el DWH ¿La alta dirección apoya el proyecto?
Intervención 2: Al iniciar el DWH ¿Los usuario entienden y apoyan el proyecto?
Intervención 3: Al iniciar el DWH ¿Los usuario requieren una amplia variedad de datos?

La primeras 3 intervenciones se refieren al inicio del proyecto y que tienen que ver con el capitulo 3 donde podemos ver que las respuestas a todas las preguntas son SI. El proyecto desde un inicio fue patrocinado por la alta dirección quien estaba convencida en iniciar las iniciativas del proceso como de tecnología de Data Warehouse. En este caso el usuario contaba con gran conocimiento de BI y sus

aplicaciones por lo que además de apoyar e impulsar el proyecto fue una gran guía, ya que los usuarios tenían gran habilidad para manipular datos y conocían las herramientas como base de datos y business objects.

Aunque el proyecto se delimito desde un inicio involucraba ir por datos a través de casi todos los sistemas del Banco por lo que la respuesta a la pregunta ¿Los usuarios requieren una amplia variedad de datos? fue si. Por lo que el camino a seguir fue crear un ambiente ODS un repositorio simple que permitió poner la información casi inmediatamente a disposición del usuario aun cuando el modelo multidimensional no estuviera listo, y clasificar claramente las necesidades de nuestros usuarios.

Es obvio que un proyecto tan grande y largo plazo sin patrocinadores y con usuarios colaborativos y entusiastas no sería sencillo y seguramente tendería a olvidarse y por ende a finalizar antes de concluir con el proyecto.

Intervención 4: Al elegir la herramienta ¿Los usuarios requieren herramientas restrictivas?

Entendiendo como herramientas restrictivas (herramientas que limitan las opciones del usuario final) aquellas que sin duda nos ayudarán a reducir la complejidad y la ambigüedad semántica, pero nos limitan las posibilidades.

Usar herramientas no restrictivas (es decir herramientas que dan una amplia gama de opciones al usuario) nos permitirán acceder a información menos estructurada, pero son más complejas de usar por los usuarios y es fácil que generen conflictos semánticos.

La respuesta dependió del perfil de usuarios, como se vio en el capitulo 5 en la pirámide de tipo de usuarios ya que se tenía una gran variedad de necesidades y de usuarios, la definición final de cada herramienta final es la que se menciona en el capitulo 7 Uso y monitoreo.

Para algunos usuarios no era lo mismo acceder a la información mediante informes realizados en herramientas específicas de BI, diseñadas para el análisis y el reporteo (Business Objects, Web), que hacerlo directamente con sentencias SQL. Los usuarios que usaban SQL podían acceder a cualquier posibilidad que exista en el DWH. Mientras que los que accesaban a información semiestructurada perdían la posibilidad de tomar cualquier dato pero obtenían simplicidad de uso y no se enfrentaban a problemas de ambigüedad semántica (esto último se consigue al colocar una capa de metadato en la herramienta de explotación).

Intervención 5 : Antes de iniciar el DWH ¿Los usuarios entienden la relación entre la información del DWH y la ejecución del DWH y la ejecución de sus tareas de negocio?

Intervención 6 : Antes de iniciar el DWH ¿Existe uno o varios usuarios finales expertos "power user"?

Intervención 7: Antes de iniciar el DWH ¿Los usuarios perciben que el departamento de IT les da soporte para resolver sus tareas?

Estas últimas intervenciones se refieren a la importancia de asegurarse de que el usuario use y aplique la información dentro de DWH, si después de todo el esfuerzo para construir un proceso DWH el usuario no sabe como usar el DWH y por ende no sabe como aplicar la información entonces los usuarios buscaran la información que si conocen y dominan como algún listado de los sistemas combinado con un Excel que otra área le proporciona. Lo cuál provocará que el DWH caiga en desuso y en un futuro corto en el olvido.

Para nuestro caso como ya lo comentaba el usuario jugo un papel importante definiendo y validando la información en cada etapa y usando la información desde una etapa muy temprana cuando se pobló la base de datos. Los usuarios tenían muy clara la visión de lo que requería el negocio además de que estaban familiarizados con los datos fuentes que alimentaba el almacén de datos, en pocas palabras si contábamos con usuarios expertos o denominados "power user". Estos usuarios explotaron el DWH y pronto como se menciona en el capitulo 7 los resultados que obtuvieron llamaron la atención de otros usuarios y pronto hubo un crecimiento de usuarios que empleaban el DWH. Otro rol que jugaban fue el de darle mantenimiento continuo a los modelos para dar respuesta a las necesidades cambiantes del Negocio, es decir, provocaban la evolución de los modelos y aplicaciones para mantenerlas actualizadas.

Aunque para las diversas áreas de IT fue difícil incorporarse a una nueva cultura de DWH pronto los primeros resultados los convencieron por completo, con lo cual se logro un ambiente de colaboración aceptable, sin embargo, la falta de implementación de todo lo descrito en el capitulo 8 sobre todo en el tema de definición de roles provoca cierta guerra de intereses que no logra crear el ambiente adecuado de colaboración, para acelerar todas las iniciativas de BI.

Para todos lo lectores que como yo estamos en campo de Ingería en computación deben considerar:

"Es un hecho entonces que el papel del ingeniero en computación, en este tipo de proyectos no se queda sólo con labor meramente técnica o tecnológica. Si se considera a un usuario de toma de decisiones el cual no puede tomar las decisiones correctas sin la información adecuada, a su ves el ingeniero en computación no puede desarrollar una aplicación de apoyo a la gestión sin tener los conocimientos de toma de decisiones adecuadas ó dicho de otra manera involucrarse con el Negocio de su usuario para brindarle el resultado correcto que los llevará a ambos a proyectos y procesos exitosos".

10. REFERENCIAS BIBLIOGRÁFICAS

Datos de catalogación bibliográfica por autor.

1. W. H. Inmon
Building the Data Warehouse
Second Edition, United States of America : Wiley Computer Publishing - John Wiley & Sons, Inc, 1996
ISBN : 0471-14161-5
Número de páginas : 401

2. Dyché, Jill
E-Data: transformando datos en información con Data Warehousing.
1ra. Edición Buenos Aires : Prentice Hall, 2001.
Traducción de : Esteban Flamini
ISBN : 987-9460-14-6
Número de páginas : 374

3. Ralph Kimball
The Data Warehouse Toolkit
First Edition, United States of America : Wiley Computer Publishing - John Wiley & Sons, Inc, 1996
ISBN : 0-471-15337-0
Número de páginas : 388

4. Informix-IMB
Informix Guide to Data Warehousing Desing and implementation
Data Warehousing Section II
Building a Dimentional Data Model Chapter 6
Implementing a Dimensional Data Model Chapter 7

APÉNDICE 1: Conceptos Base de Datos Relacional

Una base de datos relacional es una base de datos en donde todos los datos visibles al usuario están organizados estrictamente como tablas, y en donde todas las operaciones de la base de datos operan sobre éstas tablas. El modelo relacional era un intento de simplificar la estructura de las bases de datos. Eliminaba las estructuras explícitas padre / hijo de las base de datos, y en su lugar representaba todos los datos. Es más fácil describir una definición más informal de una Base de Datos Relacional.

La definición está destinada específicamente a eliminar estructuras tales como los punteros incorporados de una base de datos jerárquica o en red. Un DBMS relacional puede representar relaciones padre / hijo, pero éstas se representan estrictamente por los valores contenidos en las tablas de la base de datos.

Tablas:
El principio de organización de una base de datos relacional es la tabla, una disposición rectangular fila / columna de los valores de datos. Cada tabla de una base de datos tiene un nombre de tabla único que identifica sus contenidos. Para cada columna de una tabla, todos los valores de esa columna contienen el mismo tipo de datos. Una tabla tiene al menos una columna.

Clave primaria:
Puesto que las filas de una tabla relacional no están ordenadas, no se puede seleccionar una fila específica por su posición en la tabla. No hay "primera fila", "última fila", o "decimotercera fila" de una tabla.

En una base de datos relacional bien diseñada cada tabla tiene una columna o combinación de columnas cuyos valores identifican unívocamente cada fila en la tabla. Esta columna (o columnas) se denomina Clave Primaria de la tabla.

La clave primaria tiene un valor único diferente para cada fila de una tabla, de modo que no hay dos filas de una tabla con clave primaria que sean duplicados exactos la una de la otra. Una tabla en donde cada fila es diferente de todas las demás se llama una relación en términos matemáticos. El nombre "Base de Datos Relacional" proviene de este término, ya que las relaciones (las tablas con filas distintas) son el corazón de una base de datos relacional.

Relaciones:
Una de las principales diferencias entre el modelo relacional y los modelos de datos primitivos es que los punteros explícitos, tales como las relaciones padre / hijo de una base de datos jerárquica, están prohibidas en las bases de datos relaciónales. Obviamente estas relaciones siguen existiendo en una base de datos relacional, representada por valores de datos comunes almacenados en las dos tablas. Todas las relaciones de una base de datos relacional están representadas de este modo.

Claves Foráneas:
Una columna de una tabla cuyo valor coincide con la clave primaria de alguna otra tabla se denomina una Clave Foránea. Lo mismo que una combinación de columnas puede servir como clave primaria de una tabla, una clave foránea puede ser también una combinación de columnas. De hecho, la clave foránea será siempre una clave compuesta (multi-columna) cuando referencia a una tabla con una clave primaria compuesta. Obviamente el número de columnas y los tipos de datos de las columnas en la clave foránea y en la clave primaria deben ser idénticos unos a otros.

1. Normalización de Datos

¿Qué es Normalización de Base de Datos?

La normalización es esencialmente el proceso de tomar una tabla ancha o con muchas columnas pero de pocas filas y de reajustarla como varias tablas estrechas con pocas columnas pero más filas. Un diseño correctamente normalizado permite que usted utilice el espacio de almacenaje eficientemente, que elimine datos redundantes, que reduzca o que elimine datos contrarios, y que facilite la carga del mantenimiento de los datos.

Formas de normalización
Los teóricos de la base de datos relacional han dividido la normalización en varias reglas llamadas *las formas normales*.

- **Primera Forma Normal**
  No hay grupos repetitivos
- **Segunda Forma Normal**
  Ningún atributo de una columna no - clave depende de una porción de la clave primaria.
- **Tercera Forma Normal**
  Ningún atributo depende de otros atributos de una columna no - clave.

Además, para una base de datos que está en la segunda forma normal, debe también estar en la primera forma normal, y para que una base de datos esté en la tercera forma normal, debe cumplir los requisitos para la primera y segunda forma normal.
Hay también formas adicionales de normalización, pero éstas se aplican raramente. De hecho, puede, ocasionalmente, ser práctico violar incluso las primeras tres formas de normalización.

Primera Forma Normal
No hay grupos repetitivos

Lo que estamos buscando son grupos de columnas que se repiten. El propósito de esto es reducir el ancho de la tabla. Hacemos esto tomando los grupos de columnas y haciendo una tabla nueva donde ésta definida por las columnas repetidas. Ahora en vez de tener columnas adicionales tendremos filas adicionales en otra tabla.
Ok , así que ¿cuál es un grupo que se repite? Miremos las columnas de la tabla alumnos:

- Nom_alumno
- Tel_alumno
- Dir_alumno
- Cp_alumno
- Edo_alumno
- Nom_asesor
- Tel_asesor
- Cve_curso_D1
- Des_D1
- Nom_Prof_D1
- Tel_Prof_D1
- Cve_curso_D2
- Des_D2
- Nom_Prof_D2
- Tel_Prof_D2

En el ejemplo, las columnas para la información del curso se han duplicado para permitir que el alumno tome dos cursos. El problema ocurre cuando el estudiante desea tomar tres o más cursos. Mientras que usted podría ir al final y agregar cve_curso_d3, etc., a la tabla, la solución apropiada es mover el grupo que se repite de columnas a otra tabla.

Si ud. tiene un conjunto de columnas en una tabla con nombres que terminan en números: *xx1, xx2, xx3,* etc., está claro que hay una señal de alerta de repetición de grupos de una tabla. Claro, aplican excepciones.

La nueva tabla normalizada será:

Alumnos	**Alumnos_Cursos**
Cve_Alumno Nom_alumno Tel_alumno Dir_alumno Cp_alumno Edo_alumno Nom_asesor Tel_asesor Cve_curso	*Cve_Alumno* *Cve_curso* Des_curso Nom_Prof Tel_Prof

Primera Forma Normal Las claves primarias se muestran en rojo.

Segunda Forma Normal
Ningún atributo de una no-clave depende de una porción de la clave primaria.

La segunda forma normal realmente sólo aplica a las tablas donde las claves primarias están definidas por dos o más columnas. La esencia es que si existen columnas las cuales se puedan identificar por sólo una parte de la clave primaria, éstas deben estar juntas en su propia tabla.

Alumnos	**Alumnos_Cursos**	**Cursos**
Cve_Alumno Nom_alumno Tel_alumno Dir_alumno Cp_alumno Edo_alumno Nom_asesor Tel_asesor Cve_curso	*Cve_Alumno* *Cve_curso*	*Cve_curso* Des_curso Nom_Prof Tel_Prof

Segunda Forma Normal Las claves primarias se muestran en rojo.

Lo que hicimos fue mover toda la información de los cursos a su propia tabla *Cursos*.

Tercera Forma Normal
Ningún atributo depende de otros atributos de una columna no - clave.

Esto significa que las columnas en la tabla deben contener solamente información sobre la entidad definida por la clave primaria. Las columnas en la tabla deben

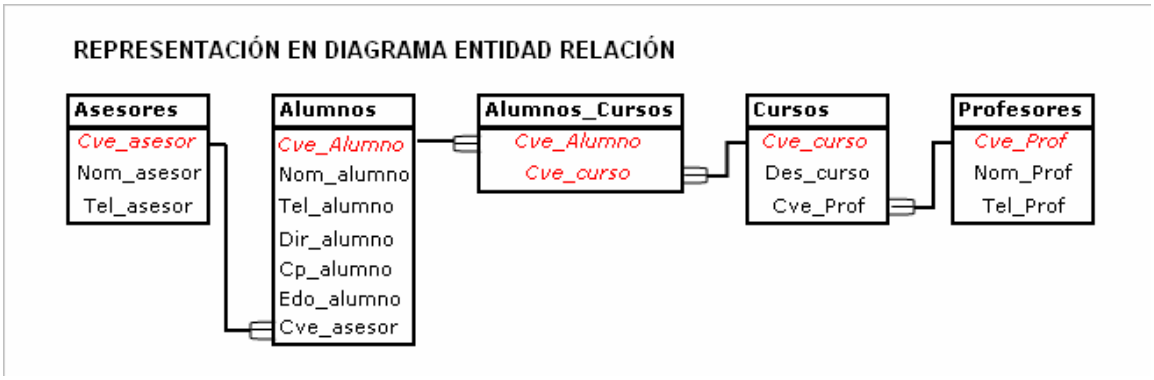
contener datos acerca de una sola cosa. Esto es realmente una extensión de la segunda forma normal, ambas son usadas para mover columnas a tablas propias.

Para terminar la normalización, tenemos que buscar columnas que no sean dependientes de la clave primaria. En la tabla *Alumnos* tenemos dos datos sobre el Asesor del estudiante: el nombre y el número de teléfono, ninguno de los dos deben estar en la tabla de alumnos.

Completemos la normalización de este ejemplo:

Alumnos	**Asesores**	**Profesores**	**Alumnos_Cursos**	**Cursos**
Cve_Alumno Nom_alumno Tel_alumno Dir_alumno Cp_alumno Edo_alumno Cve_asesor	*Cve_asesor* Nom_asesor Tel_asesor	*Cve_Prof* Nom_Prof Tel_Prof	*Cve_Alumno* *Cve_curso*	*Cve_curso* Des_curso Cve_Prof

Tercera Forma Normal Las claves primarias se muestran en rojo.



APÉNDICE 2: Data warehouse y esquemas de fragmentación.

Data warehouse: Desde el punto de vista de un DBA, data warehouse se refiere a una base datos que contiene grandes cantidades de datos históricos almacenados. Los datos son almacenados en series de fotos a lo largo del tiempo; cada registro representa datos en un punto específico en el tiempo. Estas fotografías de datos en diferentes puntos en el tiempo permiten a los usuarios reconstruir la historia y además realizar comparaciones entre diferentes periodos de tiempo.

Uno de los principales beneficios que provee data warehouse es la facilidad de acceder y analizar una gran cantidad de información almacenada previamente.

Sin embargo, así como uno de los principales beneficios que se obtienen es tener acceso a una gran cantidad de datos históricos, existe un gran problema que es cómo mantener actualizados esta gran cantidad de datos de la manera más eficiente.

Este documento pretende entregar al lector una serie de definiciones y técnicas que pueden ser utilizadas en uno de los principales problemas que se tienen cuando se está implementando u operando un data warehouse: el proceso de carga/descarga masiva de datos.

Sin profundizar mucho en el tema de la planeación, cuando se está concibiendo un data warehouse, una de las preguntas a resolver entre muchas otras es:

- ✓ Historia que se va a almacenar (h),
- ✓ Granularidad de la historia a almacenar (gh).
- ✓ Periodo de actualización (pa).

Utilizando solamente estos elementos del tiempo se puede modelar el almacenamiento de datos en un data warehouse en función de estos tres conceptos.

Ejemplo:Si tenemos una tabla que pertenece a un data warehouse donde la historia a almacenar es de tres años, la granularidad de los datos a almacenar es mensual y el periodo de actualización es mensual entonces la siguiente pregunta a resolver es ¿cuánto espacio se utiliza por tabla (considerando datos, índices y un 20% de crecimiento anual)?. Continuando con el ejemplo supongamos que la respuesta es 10 GB/mes.

En este punto estamos en posibilidad de calcular el espacio que requiere esta tabla para almacenar los tres años de historia requeridos. En términos generales sería:
10 GB/mes*24 meses/1 año*3 años=720 GB.

Durante los primeros 36 meses de carga no se tendría ningún problema en la carga debido a que en cada mes existiría un "cajón" previamente reservado para depositar los datos del mes que corresponda; claro esta que esto es cierto solamente si ni se dispone de datos históricos a cargar. El problema se presenta cuando se intente cargar el mes número 37; en este punto ya no hay disponible un "cajón" vacío por lo que será necesario diseñar un proceso de reutilización de espacio.

Otro problema potencial que se puede presentar durante una carga de datos en una tabla con historia es el tiempo de carga en cada actualización. En este punto se debe diseñar un proceso tal que el tiempo del proceso de carga de una tabla histórica no dependa de la historia de la tabla ni de los índices que tenga.

¿Qué es la fragmentación?
Fragmentación en un servidor de base de datos es una propiedad que permite controlar donde se almacenan los datos a nivel de tabla. La fragmentación permite definir grupos de registros o índices en una tabla de acuerdo a algún algoritmo o esquema.

En el caso de IBM informix XPS este grupo de datos se puede almacenar en un fragmento (o partición); una partición es un dbspace asociado a un disco físico. Para realizar esto se pueden utilizar sentencias SQL.

Al concepto de agrupar bloques de registros o índices en fragmentos se le llama esquemas de distribución. A la relación que existe entre los esquemas de distribución y un grupo de dbspaces se le conoce como estrategia de fragmentación.

Desde la perspectiva del cliente no existe diferencia entre una tabla fragmentada y una no fragmentada. Lo mismo ocurre con las aplicaciones dirigidas al usuario final, estas no requieren ninguna modificación si acceden a tablas fragmentadas o no fragmentadas.

¿Porqué utilizar fragmentación?

Considere utilizar esquemas de fragmentación cuando se requiere:
- ✓ Mejorar el tiempo de respuesta hacia el usuario final.
- ✓ Mejorar los niveles de concurrencia,
- ✓ Mejorar la disponibilidad,
- ✓ Mejorar los esquemas de respaldo y recuperación de datos,
- ✓ Mejorar el proceso de carga/descarga masiva de datos.

Dependiendo de cuál o cuales son los objetivos que se quieren perseguir, la estrategia de fragmentación puede variar en cada caso; cuando finalmente se haya decidido por la estrategia de fragmentación adecuada se debe tomar en cuenta que la fragmentación a nivel de base de datos requiere actividades de monitoreo y administración adicionales.

Esquemas de distribución para fragmentación de tablas
Un esquema de distribución es un método que el servidor de base de datos utiliza para distribuir grupos de datos o índices en fragmentos. Informix XPS soporta los siguientes tipos de esquemas de distribución:

- Basados en expresión (*expression based*): Con este tipo de esquema de distribución los registros se almacenan en el mismo fragmento si tienen el mismo valor resultado de la expresión definida previamente, la cual puede ser una regla de rankings o alguna cualquier expresión matemática. Adicionalmente se puede especificar un fragmento remainder, donde se almacenarán todos los registros que no cumplan con ningún criterio.
- Round robin (*round robin*): Con este esquema de distribución los registros se insertan de manera uniforme en cada uno de los fragmentos especificados. El servidor de base de datos define internamente la regla a utilizar.
  Para declaraciones *insert* el servidor de base de datos puede utilizar funciones *hash* con un número aleatorio para determinar el fragmento donde alojará cada registro. En cursores de *insert*, el servidor de base de datos aloja el primer registro en un fragmento al azar, el segundo registro en el siguiente fragmento secuencial y así sucesivamente. Si uno de los fragmentos se llena, el proceso no se detiene, el manejador de base de datos se lo salta y continúa con la inserción.

- Range distribution:con este esquema de distribución se asegura el que los registros se distribuyan a través de un grupo de *dbspaces* definido. Para realizar esta distribución el servidor de base de datos se basa en valores mínimos y máximos que el usuario previamente especifica. Informix recomienda utilizar este esquema de distribución cuando los datos son tanto densos como uniformes.

- (System Defined Hash): este esquema utiliza una regla definida por el sistema de manera interna que distribuye los registros con la finalidad de balancear la carga, (dejar el mismo número de registros en cada fragmento).

- (Hybrid):con este esquema de distribución se combinan dos esquemas en realidad. El esquema primario selecciona el *dbslice*. El esquema secuendario aloja los registros en un dbspace especifico (el cual pertenece al dbslice)

APÉNDICE 3: Nociones Básicas de SQL embebido en C

1. Introducción

La idea básica del trabajo con SQL embebido en lenguajes tradicionales es escribir un programa que manipule la base de datos con SQL usando las estructuras de control y variables del lenguaje tradicional.

En este documento, nos concentramos en SQL embebido en C.

2. El Enfoque Embebido

En esta sección analizaremos algunos de los problemas que se presentan al trabajar con SQL embebido. Este análisis no pretende ser detallado sino más bien una reseña básica de los problemas y sus soluciones en el estándar.

2.1 Compilación

Al trabajar con SQL embebido en C escribimos un programa con instrucciones en dos lenguajes distintos: C y SQL. Por lo tanto, será necesario compilar los programas de una forma diferente cuando trabajamos con SQL embebido en C que cuando trabajamos solo con C.

Todas las sentencias SQL deberán comenzar con las palabras **EXEC SQL** y en vez de ejecutar el compilador de C, ejecutamos el precompilador de SQL embebido. En el RDBMS PostgreSQL, el precompilador se llama **ecpg**.

La sintaxis del comando **ecpg** correspondiente al precompilador es la siguiente:

ecgp [opciones] archivos

Por detalle sobre las opciones consultar la documentación del comando

Ejemplo compilación: Para compilar el archivo *prog1.pgc* creando un ejecutable con nombre *prog1* escribimos la siguiente secuencia de comandos:

1. ecpg prog1.pgc: En el primer paso estamos precompilando el código que contiene las sentencias SQL embebidas en C.
2. cc -I/usr/local/pgsql/include -c prog1.c: El paso siguiente crea el código objeto (para lo cual necesito los archivos de encabezado de ECPG que se encuentran en usr/local/pgsql/include
3. cc -o prog1 prog1.o -L/usr/local/pgsql/lib –lecpg: El último paso genera el ejecutable, para lo cual precisa linkear con la biblioteca libecpg ubicada en /usr/local/pgsql/lib

Ejemplo de código

A continuación presentamos un ejemplo del código C con SQL embebido que corresponde al programa *pepe.ec*.

```
(1)  #include <stdio.h>

(2)  EXEC SQL include sqlca;

(3)  main()
     {
```

```
(4)    EXEC SQL BEGIN DECLARE SECTION;
(5)    int cant;
(6)    EXEC SQL END DECLARE SECTION;

(7)    EXEC SQL connect to 'dbventas';

(8)    EXEC SQL
           select count(*) into :cant
           from   cliente;

(9)    printf(``La cantidad de tuplas de la tabla Cliente es %d \n'', cant);

   }
```

2.2 Comunicación con el DBMS

Comencemos a observar más detenidamente el programa anterior. En particular las líneas (4) a (6).

Estas instrucciones declaran una variable de comunicación entre el DBMS y el programa. Siempre que se utilice esa variable en una instrucción SQL se debe escribir ``:'' delante del nombre de la variable para que el pre-compilador interprete el símbolo como una variable de programa. Si se utiliza en cualquier instrucción de C no se debe escribir más que el nombre de la variable tal cual fue declarado.

Podemos utilizar la declaración **DECLARE SECTION**, para declarar más de una variable.

```
EXEC SQL BEGIN DECLARE SECTION;
       int hostint;
       long hostlong;
       double hostdbl;
       char hostarr[80];
EXEC SQL END DECLARE SECTION;
```

Finalmente, hacemos notar que se pueden escribir tantas DECLARE SECTION como sean convenientes.

2.3 Correspondencia entre tipos de C y tipos de SQL

Para utilizar correctamente las variables de comunicación, es necesario conocer las correspondencias entre los tipos de C y los tipos de SQL. Dicha correspondencia es presentada en tabla siguiente:

SQL	***Lenguaje C***
CHAR(n)	char[n + 1]
CHARACTER(n)	
SMALLINT	short int
INTEGER	long int
INT	
DECIMAL	dec_t o struct decimal
DEC	

NUMERIC	
SMALLFLOAT	float
REAL	
FLOAT	double
DOUBLE PRECISION	
MONEY	dec_t o struct decimal
SERIAL	long int
DATE	long int
DATETIME	dtime_t o struct dtime
INTERVAL	intrvl_t o struct intrvl

La declaración de las variables en los programas tales como *pepe.ec* arriba, se realiza según los tipos de C.

Los tipos **dec_t**, **dtime_t** e **intrvl_t** están definidos en diferentes **.h** que pueden ser incluidos en nuestros programas mediante la directiva:

```
EXEC SQL include nomarch
```

También podemos definir estructuras y tipos de usuarios con typedef. Al utilizar variables de dichos tipos, podemos hacer trabajar tanto con las componentes como con las estructuras como un todo. En realidad, siempre que aparece una variable de tipo estructura, el pre-compilador expande a la lista de campos correspondientes. Dada la siguiente declaración:

```
EXEC SQL struct cliente_t {
        int nro_cli;
        char nombre[32];
        char apellido[32];
        } tupla_cli;
```

La instrucción:

```
EXEC SQL insert :tupla_cli into cliente;
```

es equivalente a:

```
EXEC SQL insert into cliente
        values (:tupla_cli.nro_cli, :tupla_cli.nombre, :tupla_cli.apellido);
```

Por último, cabe notar que Informix las palabras claves ``**EXEC SQL**'' se pueden substituir por ``**$**''.

2.4 Impedance Mistmatch

SQL es un lenguaje para manipulación de conjuntos de tuplas, donde cada tupla es similar a un registro (**record** o **struct**) de un lenguaje tradicional.

En general, C maneja datos individuales y no conjuntos. Para manejar conjuntos en C tenemos que realizar alguna forma de iteración que permita manejar datos individuales.

Debido a la diferencia de *granularidad* con que trabajan ambos lenguajes, es necesario que el pre-compilador provea un mecanismo para resolver este problema. Para ello se introduce la noción de **cursor**.

Un cursor es básicamente un puntero a un área de memoria donde está almacenado el resultado de una consulta. Sin embargo, no es un puntero común del lenguaje de programación (en nuestro caso de C). Debe manipularse con un conjunto de operaciones especiales provistas. Algunas de estas operaciones son: **Declare, Open, Fetch**. A continuación damos una breve explicación del significado de cada operación.

- **Declare.** Declara un cursor para poder acceder al resultado de una instrucción SQL. Por ejemplo,

```
EXEC SQL declare c_tabla cursor for
                    select *
                    from   tabla;
```

- **Open.** Ejecuta la consulta asociada al cursor y hace accesible el resultado por medio del cursor. El cursor queda posicionado antes de la primera tupla del resultado. Por ejemplo,

```
EXEC SQL open c_tabla;
```

- **Fetch.** Posiciona el cursor en la siguiente tupla. Luego de un open, es necesario ejecutar esta operación para poder acceder a la primera tupla. Por ejemplo,

```
EXEC SQL fetch c_tabla into :struct_tupla_tabla;
while (sqlca.sqlcode == 0) {
      print_struct(struct_tupla_tabla);
      EXEC SQL fetch c_tabla into :struct_tupla_tabla;
}
```

La operación **fetch** permite posicionarse en cualquier tupla del resultado. Esto se logra agregando alguna de las siguientes opciones antes del nombre del cursor:

- **Previous.** Posiciona el cursor en la tupla anterior a la tupla que está en este momento.
- **First.** Posiciona el cursor en al primer tupla del resultado.
- **Last.** Posiciona el cursor en la última tupla accesible.
- **Relative *n*.** Posiciona el cursor en la tupla *n* a partir de su posición actual. Si n es negativo, entonces estamos especificando una posición anterior a la de la tupla actual.
- **Absolute *n*.** Posiciona el cursor en la tupla de ordinal *n* del resultado.

3. La Captura de Mensajes del DBMS

Después de la ejecución de una instrucción SQL, podemos informarnos del estado de la ejecución o bien por un test explícito utilizando la estructura SQLCA (*SQL Communication Area*) o bien por un test implícito utilizando la instrucción WHENEVER. Estas dos formas de manejo de errores son descriptas brevemente a continuación.

3.1 La Estructura SQLCA

En el ejemplo de la operación **fetch** de la sección 2.4, se utiliza la siguiente instrucción:

```
while (sqlca.sqlcode == 0)
```

La variable de estructura llamada **sqlca** es el mecanismo que se utiliza para recibir mensajes del DBMS. La estructura se encuentra declarada en el archivo **sqlca.h** y debe ser incluida en todos los programas en los que queremos controlar el resultado de la ejecución de operaciones por parte del DBMS. Un ejemplo de esta inclusión se encuentra en la línea (2) del programa *pepe.ec* de la sección 2.1.

Si la última operación que se envió a ejecutar al DBMS se terminó sin ninguna característica especial, entonces el valor de la variable **sqlca.sqlcode** es 0. En caso de que esa ejecución genere un error, el valor de la variable **sqlca.sqlcode** será negativo y corresponde a un código de error. El valor de la variable **sqlca.sqlerrm**, corresponde a un *string* que representa el mensaje de error que emite el DBMS. Cuando el DBMS ejecuta una instrucción **select** sobre una tabla que no tiene tuplas o una instrucción

fetch que pretende ``pasarse'' de la última tupla, el valor de la variable **sqlca.sqlcode** es aquel de la constante **SQLNOTFOUND** . **SQLNOTFOUND** es una constante definida en el archivo **sqlca.h** que debe ser incluido en caso de utilizar dicha constante.

La variable de estructura sqlca tiene más campos que los presentados aquí. Esos campos están destinados a recuperar datos tales como el último identificador de tupla generado o determinar exactamente que ``warning'' fue el que se generó por parte del DBMS. Se aconseja consultar el manual de SQL embebido provisto con el DBMS en caso de necesitar estos campos.

3.2 La Instrucción WHENEVER

La instrucción **whenever** funciona como un manejador de excepciones primitivo. En cualquier parte de nuestro programa podemos utilizar la instrucción **whenever**.

La instrucción:

```
EXEC SQL whenever sqlerror goto :unaetiqueta;
```

tiene el siguiente efecto. En el momento que se ejecuta una instrucción SQL que produce un error, la ejecución continuará en la instrucción identificada por etiqueta *unaetiqueta*.

En lugar de **sqlerror**, otras declaraciones son posibles tales como **not found** o **sqlwarning**. **not found** especifica que cuando se ejecuta un *select* sobre una tabla sin tuplas o un fetch que pretende referirse a una tupla que está después de la última tupla del resultado, se continuará la ejecución en la instrucción identificada por la etiqueta. Con **sqlwarning**, se continuará la ejecución en la instrucción identificada por la etiqueta cuando se genere un ``warning''.

Una vez que se ejecuta el **whenever-goto**, éste queda activo hasta que se ejecute otro **whenever-goto** con la misma condición o bien se ejecute una instrucción **whenever <condición> continue**.

Este mecanismo de manejo de errores tiene sus problemas. Al trabajar con un lenguaje estructurado, no se puede salir de una función usando un **goto**. Por ello, es aconsejable elegir alguna de las siguientes guías:

1. Redefinir la etiqueta de un **whenever** en cada función que contenga instrucciones SQL que puedan activarlo.
2. Definir un **whenever** con su etiqueta correspondiente en cada función que contenga instrucciones SQL que pueda activar el **whenever**.

IMPORTANTE!: la primera instrucción en un trozo de instrucciones identificado por una etiqueta, debe ser un **whenever-continue** con la misma condición que activó la etiqueta. De lo contrario, si hay instrucciones SQL en el trozo de instrucciones identificado por la etiqueta que son capaces de generar las misma condición que nos llevó hasta aquí, el programa entrará en *loop*.

4. Conclusión

SQL embebido en C no es todo lo práctico que debiera ser. Sin embargo, permite en general, obtener mejores desempeños de ejecución que los lenguajes de cuarta generación. Esta observación general sugiere entonces que SQL embebido en C sea utilizado principalmente cuando existen restricciones de performance que no se logran con otros enfoques de más alto nivel.

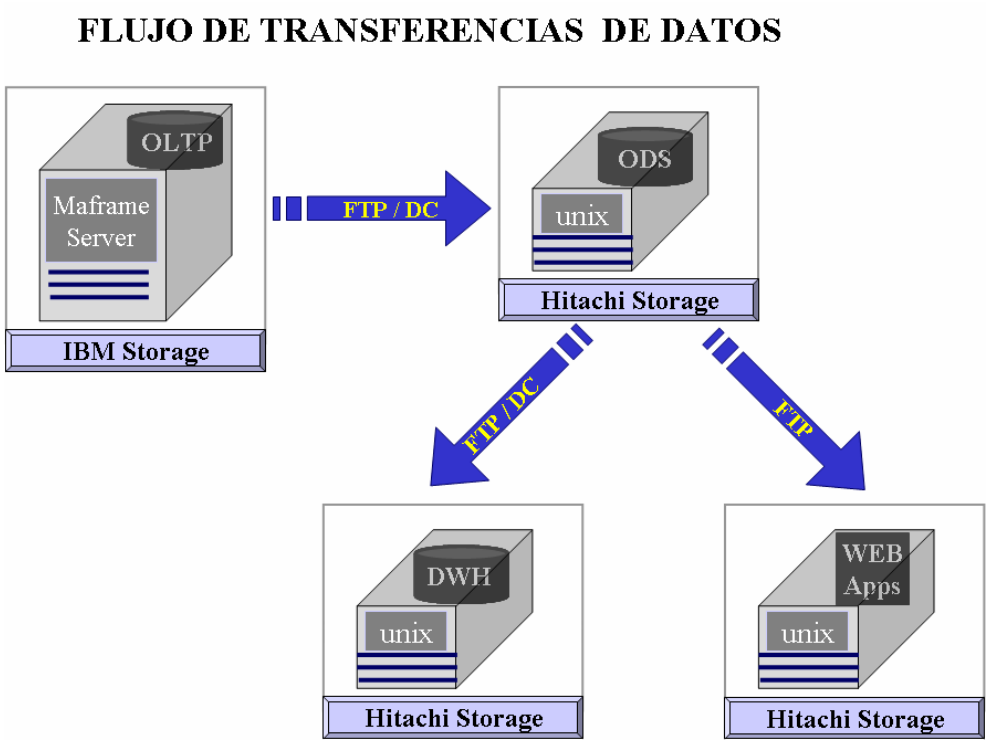
En este documento, sólo se presentó el mecanismo general. Existe un conjunto bastante grande de excepciones y casos particulares para cada aspecto presentado, así como extensiones que cada DBMS agrega. Es importante revisar los manuales del DBMS con el que se trabaja para buscar las aclaraciones de las excepciones y extensiones que correspondan. En el apéndice A se proveen referencias a la documentación de Informix.

A. Manuales de PostgreSQL sobre SQL Embebido
La documentación de PostgreSQL se encuentra disponible en la dirección:

http://http://www.postgresql.org/docs/8.0/static/

APÉNDICE 4: El problema de transferencia, solución a un problema crítico.

El crecimiento del volumen de información, el 80% de los insumos requeridos por el data warehouse provienen del mainframe, y el data warehouse se encuentra dividido en dos equipos unix (sun10k). El problema a resolver era como realizar la transferencia de grandes volúmenes de información sin afectar la red de comunicación del Banco. La siguiente lámina muestra la arquitectura propuesta para la transferencias de datos para el data warehouse.



Esta arquitectura inicialmente cubría las expectativas de funcionalidad pero en cuanto la demanda de archivos hacia los sistemas fuente y hacía las diferentes aplicaciones a usuarios finales creció, los protocolos de transferencia utilizados hasta entonces, que básicamente eran direct connect y FTP empezaron a generar problemas de saturación de la red de comunicación de datos del grupo. Errores al interrumpirse la comunicación

y no generar códigos que permitieran conocer el estado de las transferencias, tiempos excesivos de transferencias (en procesos de actualización mensual de datos se tuvieron picos de hasta 15 o más horas solamente para la transferencia de insumos hacia la plataforma del data warehouse).

Para corregir este problema se solicito al equipo de tecnología encontrar una solución al desempeño de la plataforma data warehouse. El equipo de tecnología de DWH probó diferentes alternativas centrándose finalmente en alguna tecnología de almacenamiento de datos que permitiera compartir datos entre diferentes plataformas sin necesidad de que esta viajara por la red de datos. Durante algunos meses se realizaron pruebas con diferentes proveedores de software hasta que se encontró la solución.

Utilizar una tecnología de almacenamiento de información que permitiera compartir datos entre diferentes plataformas sin necesidad de transferir dichos datos a través de las redes de comunicación.

EMC es un proveedor de sistemas de almacenamiento de datos. La solución que propone se conoce como EMC infomover que básicamente tiene dos componentes: infomover file system e infomover file transfer.

EMC infomover es una solución que combina la transferencia de archivos y la capacidad de compartirlos entre múltiples plataformas, permitiendo a los usuarios acceder a los mismos desde un centro de datos. Infomover consiste en dos componentes:

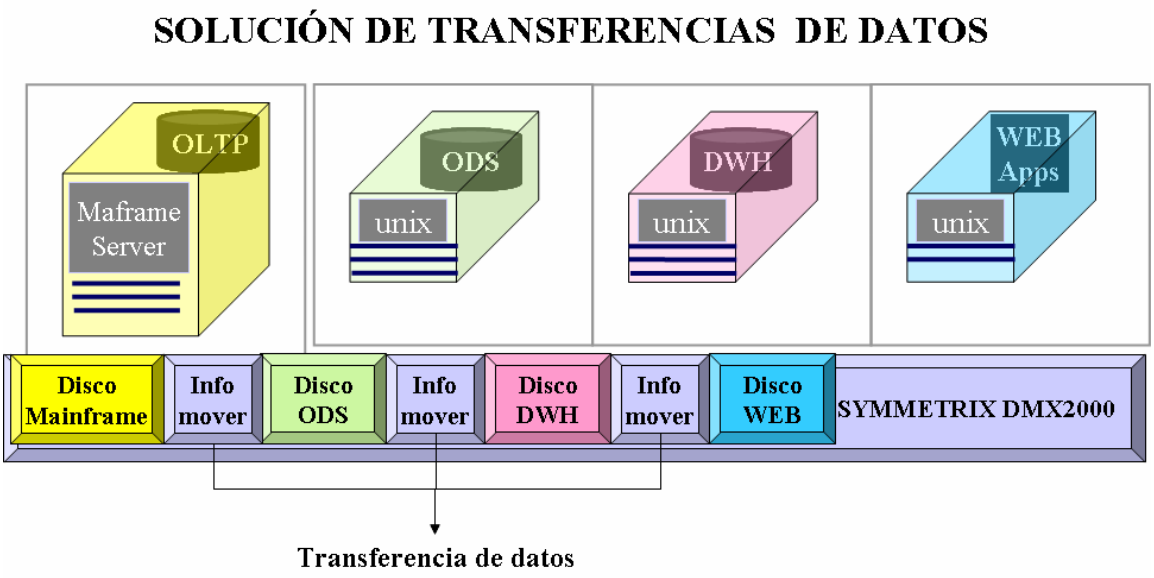
1. InfoMover File Transfer (IFT)

   InfoMover File Transfer es un componente de software que copia datos entre diferentes plataformas (VSAM-UNIX, UNIX-UNIX, UNIX-Windows, etc)a grandes velocidades. IFT le provee al usuario una funcionalidad comparable a otros protocolos de transferencia muy comunes en el mercado como lo son el file transfer program (FTP) o TCP/IP.

2. InfoMover File System (IFS)
   Es un componente de software que permite compartir datasets (paquetes de datos) entre las plataformas más comerciales como los son VSAM, no VSAM, UNIX y sistemas basados en Windows.

   Ambos componentes de infomover son propiedad de la marca EMC y son utilerías exclusivas de sus sistemas de almacenamiento que incluyen los modelos Symmetrix DMX1000, DMX2000 y los ESP (Enterprise Storage Platform) 5000 y 8000.

En la siguiente lámina se muestra un esquema de la arquitectura que resolvió el problema de la transferencia de grandes volúmenes de datos entre las diferentes plataformas que constituyen el data warehouse.



La solución consiste en tener una sola plataforma de almacenamiento de datos integrada por un arreglo de discos EMC modelo DMX2000 que tiene la capacidad de estar conectado tanto a sistemas open como a MVS.

Durante el proceso de configuración se crearon pequeñas áreas comunes entre los equipos y plataformas donde se requería tener un canal de transferencia de alta velocidad (zonas en gris en la lámina). Cuando se invoca la solución de software infomover por ejemplo para transferir un archivo entre mainframe y unix, la transferencia se realiza a través de 'paquetes de datos' cuyo paralelismo y velocidad de transferencia es configurable desde el momento del diseño físico ya que se pueden dedicar varios canales de transferencia entre esos puntos (obviamente uno de los canales de transferencia más robustos de la solución está entre los puntos mainframe-unix por el volumen de transferencia de datos que se realiza).

Con el uso creciente de la infraestructura ofrecida y el resultado de las mediciones (monitoreo) era obvio que la plataforma requería un mantenimiento de sus recursos y evolución de sus procesos y metodología.