UNIVERSIDAD DEL BÍO - BÍO
FACULTAD DE CIENCIAS EMPRESARIALES
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN Y TECNOLOGÍAS DE INFORMACIÓN

# ESTRATEGIA DE EJECUCIÓN DE SECUENCIAS DE CONSULTAS A BASES DE DATOS

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL EN INFORMÁTICA

**Alumno: Sebastián Fernández G. - Profesor Guía: Mónica Caniupán**

**Chillán, Marzo 2012**

## Resumen

En la actualidad las tecnologías han permitido resolver la necesidad de obtener información de manera eficaz, para tomar diferentes decisiones dentro de una organización. Sin embargo, esta realidad no se evidencia en todas las entidades o corporaciones debido a la falta de recursos, entre otras causas. Por ende, ésta es la principal motivación por la cual se ha llevado a cabo este proyecto.

Este proyecto implementa la optimización de consultas pertenecientes al sistema "control de la unidad de farmacia" del Centro de Salud Familiar (CESFAM) Violeta Parra, entidad dedicada a la atención primaria de salud en la provincia de Ñuble. Analizando como se ve afectado al pre-procesar las múltiples consultas que actúan sobre este sistema e implementado una nueva estrategia de ejecución de dichas consultas.


Palabras claves: optimización, base de datos, consulta.

# Tabla de Contenido

# Capítulo I: Introducción

Existe una amplia gama de servicios en Internet los cuales, en ciertos períodos de tiempo, tienden a colapsar, lo que trae como consecuencia un deterioro en los tiempos de respuestas. Por Ejemplo, el sitio de Impuestos Internos, cerca de la fecha límite fijada para declarar los impuestos. Otro ejemplo lo podemos encontrar en nuestra propia Universidad al momento de generar las actas de calificaciones.

Es posible, que sean varios los factores que expliquen el mal servicio; entre ellos podemos mencionar; la velocidad de la red, la capacidad del servidor, desempeño del software subyacente (Sistema Operativo, Sistema de Administración de Bases de Datos, entre otros). Suponiendo servicios, en los cuales se requiere procesar muchas consultas (provenientes de usuarios) y que implican acceder a una base de datos, podría ser conveniente pre-procesar el conjunto de consultas con tal de aplicar una estrategia de ejecución diferente al orden de llegada. A modo de Ejemplo, si tenemos dos consultas c1 y c2 y podemos detectar que la respuesta de c1 está incluida en la de c2, es claro que es más conveniente procesar primero c2 y luego refinar la respuesta para generar la de c1.

Nos enfocaremos en el sistema de control farmacia del CESFAM Violeta Parra en donde se ingresan una gran cantidad de datos diariamente y se realizan un gran número de consultas al momento de la obtención de reportes. Para este sistema se pretende plantear una nueva estrategia de ejecución, para el procesamiento de sus consultas. Además de analizar y mejorar algunos de sus factores que explicarían el mal funcionamiento del servicio, mejorando así los tiempos de respuestas. Cabe destacar que para elaborar una estrategia de ejecución para el procesamiento de consultas es necesario asegurarnos de que dichas consultas sean correctas y óptimas.

Al pre-procesar un conjunto de consultas de base de datos y aplicando una nueva estrategia de ejecución, para una aplicación en particular, se mejoran los tiempos de respuestas al detectar que consultas o las respuestas de algunas consultas (resultados intermedios) están contenidas dentro de otras consultas.

## 1.1 Objetivos

### 1.1.1 Objetivo General

- Elaborar o adaptar, para el servicio de "control de farmacia" del CESFAM Violeta Parra, una estrategia de ejecución de secuencias de consultas de base de datos, de tal manera de disminuir el tiempo total y amortiguar, ante una alta demanda, el deterioro del servicio.

### 1.1.2 Objetivos Específicos

- Evaluar y analizar situación actual del servicio, (sistema de control de farmacia) y los factores que influyen en él.
- Comprender las tareas de procesamiento y optimización de consultas realizadas por un sistema gestor de bases de datos relacional.
- Conocer reglas heurísticas y de transformación de expresiones del álgebra relacional y cómo aplicarlas para mejorar la eficiencia de una consulta.
- Conocer otros enfoques de optimización de consultas: optimización de múltiples consultas.
- Proponer una estrategia que mejore la situación actual del servicio que se estudia (minimización del tiempo total ante una alta demanda).
- Implementar y evaluar la estrategia propuesta.
- Obtener conclusiones respecto a las ganancias obtenidas.

## 1.2 Metodología de Trabajo

La metodología de trabajo se basa en estudio de la literatura y posterior aplicación, con las siguientes etapas:

- Identificación de la necesidad del estudio, esto conlleva un análisis de la situación actual del sistema de farmacia del CESFAM Violeta Parra e identificación de problemas.
- Propuesta y definición de los temas que se estudian.
- Revisión y estudio de la literatura técnica:
    - Extracción de información y síntesis de resultados.
- Desarrollo y aplicación de los estudios realizados, realizando experimentos y pruebas correspondientes.

### 1.2.1 Justificación de la Metodología de Trabajo

Esta es una forma de investigación que proporciona un resumen de distintos estudios existentes sobre un tema específico, utilizando para ello una revisión y estudio de la literatura técnica, evaluación crítica y síntesis de la literatura.

# Capítulo II: Descripción de la Empresa

## 2.1 Introducción

En este capítulo, se presenta una descripción del Consultorio Violeta Parra, donde se dará ha conocer sus principales características tales como; la organización, misión y visión, y finalmente se muestra una descripción de la situación actual del sistema de control de farmacia, es en este sistema que aplican las distintas técnicas de optimización por ende se trabaja con él a lo largo del proyecto.

## 2.2 Descripción General de la Organización

### 2.2.1 Institución

Razón Social : Centro de Salud Familiar Violeta Parra

Giro : Salud

Rut : 61.931.800-5

Director(a) : Pamela Zamudio Villarroel

Dirección : Francisco Ramírez #150, Chillán

Teléfono : 42 – 203624

### 2.2.2 Historia de la Organización

El Centro de Salud Familiar Violeta Parra fue construido en 1992 al costado este del Hospital Herminda Martin siendo en un comienzo un Consultorio General Urbano (CGU) convirtiéndose en CESFAM el año 2007. Sus funciones se encuentran destinadas a la atención primaria de salud y sus objetivos principales son: el fomento, producción, protección y prevención de enfermedades.

El CESFAM, es un establecimiento de Atención Primaria de la Comuna de Chillán, dependiente técnica y administrativamente del Servicio de Salud Ñuble. Funciona como unidad presupuestaria y administrativa independiente desde mayo de 1995. Su estructura física está hecha en albañilería reforzada de reciente construcción, con una superficie de 1.665, 21 MT.2 construidos y ubicado en un terreno de 4.860 [$m^2$].

Actualmente la población inscrita en el CESFAM Violeta Parra es alrededor de 77.200 personas, lo que equivale a un 43,15 % de la población de Chillán (CESFAM Violeta Parra, 2008) .

### 2.2.3 Organización

El Centro de Salud Violeta Parra se encuentra dividido en cinco sectores para brindar una mejor atención a sus pacientes los cuales son: Sector 1, Sector 2, CECOF Padre Hurtado, Sector 4, Sector 5, Sector 6 y Sector 7. La unión de éstos abarca el territorio geográfico de la ciudad de Chillán, por lo tanto, dividen a los pacientes según el sector donde habitan. Los sectores 1, 2, CECOF, 4 y 5 se limitan geográficamente de la siguiente forma:

- Límite norte: comprende desde el puente ferroviario sobre el Río Ñuble en su extremo sur, siguiendo por la rivera hasta la unión con el Río Cato, luego por la rivera sur de éste hasta el kilómetro 5, camino a Coihueco.
- Límite Sur: comprende Avda. Collín en su lado Norte, entre avenidas Brasil y Argentina, siguiendo por calle Colón lado norte hasta el límite comunal de Coihueco.
- Límite Este: con la comuna de Coihueco.
- Límite Oeste: Avda. Brasil en su vereda este, hasta la esquina con Avda. Ecuador, para seguir desde allí por la línea férrea hasta el puente Ñuble.

Lo anterior se aclara en la Figura 1: Radio de Acción CESFAM Violeta Parra.

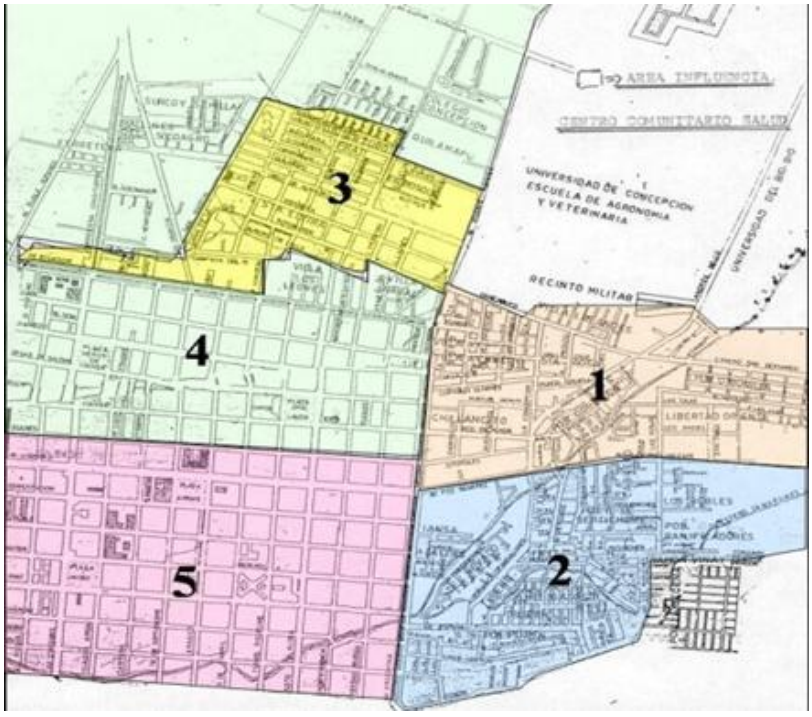



Figura 1: Radio de Acción CESFAM Violeta Parra

El Sector 6 representa a la población que se encuentra fuera del radio de acción ya mencionado y el Sector 7 constituye a la población de los sectores rurales.

En la Tabla 1: Población Inscrita Hasta el 13 de Agosto de 2010 se presenta el sector y la población que tiene inscrita en cada uno de éstos.

| Sector | Hombres | Mujeres | Total por Sector |
|---|---|---|---|
| 1 | 5.809 | 7.395 | 13.205 |
| 2 | 7.645 | 9.136 | 16.783 |
| CECOF | 2.865 | 3.366 | 6.231 |
| 4 | 5.606 | 7.448 | 13.058 |
| 5 | 4.798 | 6.102 | 10.905 |
| 6 | 5.872 | 6.682 | 12.560 |
| 7 | 609 | 686 | 1.302 |
| TOTAL | 33.204 | 40.815 | 74.019 |

Tabla 1: Población Inscrita Hasta el 13 de Agosto de 2010

En cada sector atienden enfermeras, matronas, trabajadores sociales, nutricionistas, psicólogos y técnicos paramédicos. Los odontólogos atienden de manera transversal, es decir, sin diferenciación de pacientes por sector. El organigrama de la institución se detalla en la Figura 2.

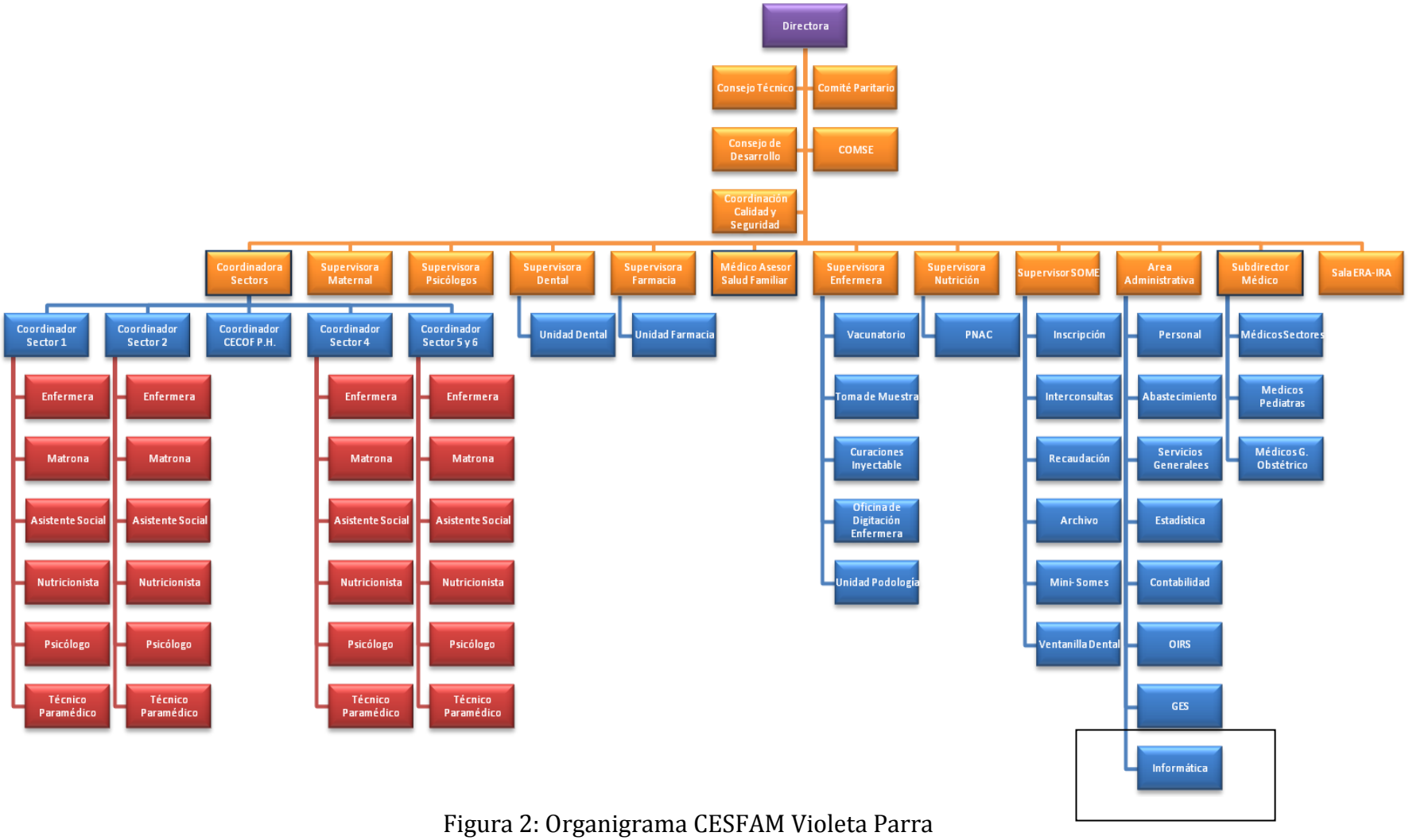



Figura 2: Organigrama CESFAM Violeta Parra

### 2.2.4 Misión y Visión

La misión de esta institución consta en satisfacer en forma efectiva y eficiente las necesidades de salud de la población beneficiaria, en coordinación con otras instituciones sociales, priorizando a los sectores más pobres y vulnerables, maximizando la accesibilidad de la población a acciones de salud humanizadas, destinadas a la promoción, fomento, prevención y recuperación de la salud de las personas y el medio ambiente en el marco conceptual de la estrategia de la Atención Primaria.

Y la visión en contribuir al bienestar de la población del sector, con el fin de lograr familias sanas, en un entorno saludable como reflejo de un satisfactorio trabajo en equipo y en red (CESFAM Violeta Parra, 2008).

### 2.2.5 Valores

- Equidad: ofrecer igualdad de oportunidades en el acceso a la atención, de acuerdo a las prioridades sanitarias.
- Participación: trabajar en equipo tanto al interior del Consultorio como con los integrantes de la red asistencial y la comunidad de Ñuble.
- Satisfacción usuaria: otorgar un trato amable y oportuno.
- Eficiencia: obtener los mejores resultados posibles con los recursos existentes.
- Efectividad: entregar medicina basada en la mejor evidencia científica disponible.

### 2.2.6 Descripción del Área de Estudio

- **Área en la cual se realizó el estudio:** Departamento de Informática (cuadro destacado de la Figura 2).
- **Descripción del área:** La unidad de Informática en el CESFAM Violeta Parra se encarga de desarrollar, mantener y dar soporte a los sistemas de información utilizados por la institución.
- **Actividades que se realizan en el área:** Aquí se planifica, analiza, diseña y construye el software solicitado por el CESFAM además de prestar mantención a las aplicaciones en operación. Se dan servicios de soporte a los computadores que existen en el CESFAM.
- **Persona encargada:** Claudio Torres Añasco, Jefe Departamento de Informática

## 2.3 Sistema de Control de Farmacia

Cada establecimiento de asistencia primaria de salud, como es el caso del "CESFAM Violeta Parra" de Chillán, está organizado en distintas unidades, que hacen posible desarrollar el trabajo de la institución y cumplir con sus objetivos. Entre estas unidades se encuentra el departamento de farmacia.

La atención farmacéutica es el conjunto de prestaciones, que se brindan a través de los servicios, insumos y medicamentos, integrados a la atención de salud del paciente, familia y comunidad. *El objetivo de la atención farmacéutica, es lograr la racionalidad en el uso de los medicamentos, mediante un adecuado suministro de estos; educando al equipo de salud y pacientes sobre su buen uso y estimulando la responsabilidad de cada persona en el uso racional de los fármacos para el cuidado del enfermo* [MINSAL, 1995]. Para lograr el objetivo anterior es necesario, contar con sistemas informáticos que apoyen la gestión de las farmacias y aporten información de control para la toma de decisiones. Dichos sistemas de gestión, fueron por muchos años llevados en forma manual, a través de planillas, libros de registro de entregas, inventario y tarjetas de dispensación de medicamentos a pacientes. En el "CESFAM Violeta Parra" se implementó el sistema de control de farmacia (como sistema informático) para a poyar a la unidad de farmacia el 01 de abril de 2011. Este sistema se implemento utilizando herramientas como: java/J2EE, hibernate, Struts 2, JQuery y MySQL.

El departamento de farmacia se organiza para atender básicamente a dos tipos de pacientes, a fin de brindar una mejor y más ordenada atención a las personas. Estos tipos de pacientes son "crónicos" y de "morbilidad". A continuación se detalla su descripción:

a) Paciente Morbilidad: Persona que acude al Centro de Salud en busca de atención médica en forma circunstancial a causa de enfermedad o sintomatología que no sufre periódicamente, o que al menos no ha sido diagnosticada como enfermedad crónica.

b) Paciente Crónico: Persona que asiste mensualmente o a intervalos regulares, a atención de salud (controles), por habérsele diagnosticado enfermedad crónica, y estar afecto a un programa de atención de enfermos crónicos y que por esta razón retira medicamentos de farmacia crónicos en forma regular y controlada.

Por otra parte el Departamento de Farmacia se preocupa de abastecer otras farmacias como son la farmacia SAPU que opera en horario extraordinario en dependencias del mismo CESFAM. También están consideradas las farmacias externas al CESFAM, como es el caso de algunas postas rurales que son abastecidas eventualmente en virtud del plan de redes asistenciales que articula el Ministerio de

Salud a través de sus Secretarías Regionales y Servicios de Salud Regionales y/o Provinciales. Por último, dentro de las farmacias externas al Consultorio que son abastecidas, está la farmacia del "CECOF Padre Hurtado" (Centro Comunitario de Salud Familiar) creado por iniciativa del CESFAM y dependiente de él en su gestión y funcionamiento.

En las farmacias de morbilidad y crónicos, laboran técnicos paramédicos, capacitados para orientar a los pacientes, sobre el uso de los medicamentos que les son entregados. Para comprender de una mejor forma el funcionamiento del control de farmacia del CESFAM Violeta Parra se muestra a gráficamente el modelo del proceso de negocio para dispensar una receta en la farmacia en la Figura 3.

Al dispensar una receta el sistema de "control de farmacia" debe llevar un control exacto de los productos retirados, preocupándose de:

- A quienes se les entregaron los productos.
- Cuando se dispenso la receta.
- Quién recetó las prescripciones.
- Control de stock de los productos.

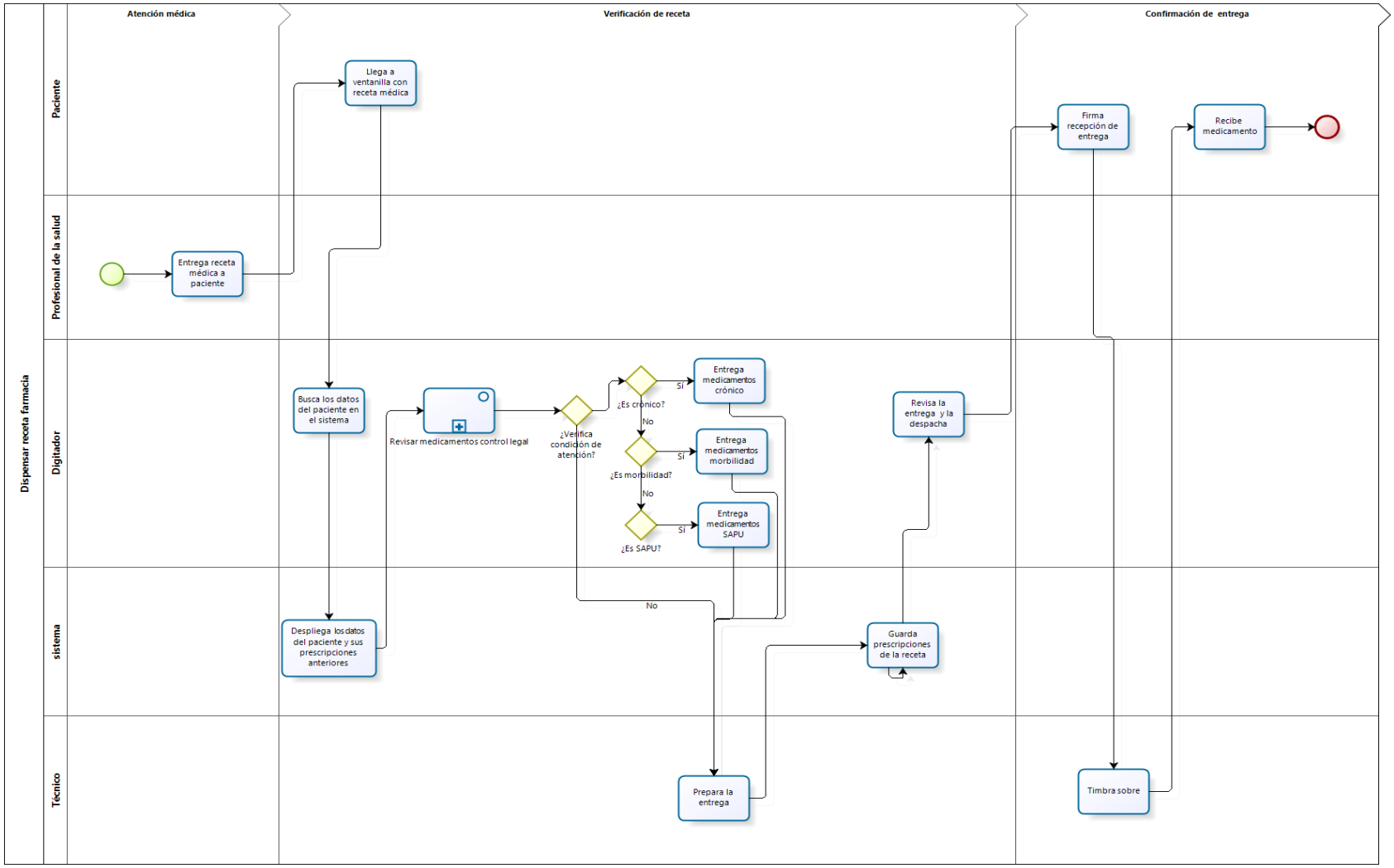



Figura 3: Modelo del Proceso de Negocio para Dispensar Receta Farmacia[1].

[1] Para obtener mayor información sobre el modelo BPMN visitar http://www.bpmn.org/

### 2.3.1 Información que Maneja el Sistema de Control de Farmacia

A continuación se lista la información más relevante que se maneja en el sistema de "control de farmacia." Además en la Figura 4 se detalla el modelo entidad relación existente en el CESFAM Violeta Parra.

- Los productos: estos son los remedios que son entregados a los pacientes.
- Los usuarios: encargados de la unidad de farmacia que hacen uso del sistema y que hacen las entregas de los productos.
- Los pacientes: pude ser un paciente con procedencia crónica, de morbilidad o SAPU y son quienes retiran los productos.
- Los profesionales: quienes prescriben los productos a los pacientes para luego ser retirados en farmacia.
- Las recetas y sus prescripciones (detalles de una receta).
- Además para la obtención de reportes e información adicional el sistema de control de farmacia interactúa con información de otros sistemas pertenecientes al CESFAM (por ejemplo el sistema de reservas de horas).

Los usuarios que utilizan este sistema son: "Administrador Farmacia" y "Técnico Farmacia" además del centro de estadísticas que hacen utilización de ella durante algunos periodos (fin de mes y fin de año principalmente).

Esta aplicación maneja una gran cantidad de datos almacenados en la base de datos. El sistema de control de farmacia comenzó a utilizarse el 01 de abril de 2011 y hasta la 01 de octubre de 2011 se han ingresado 77.225 recetas y 197.149 prescripciones con un promedio diario de ingreso de recetas y prescripciones de 635 y 2.000 respectivamente, estos nos indica la gran cantidad de datos que se deben procesar para la obtención de reportes y otras consultas además esta base de datos se mantendrá creciendo mediante el tiempo siga transcurriendo (esta información solo incluye la información respecto a recetas no incluye los productos o pacientes). A continuación en la Figura 5 se muestran un gráfico que representan la cantidad diaria de recetas y en la Figura 6 productos dispensados (prescripciones) registrados y en la Figura 7 el gráfico de datos acumulados que representan las recetas diariamente dispensadas y en la Figura 8 el gráfico que representa las prescripciones acumuladas en el mes de junio (por lo general la cantidad de datos ingresados mes a mes no presentan una gran variación por lo que este mes se escogió de forma arbitraria).

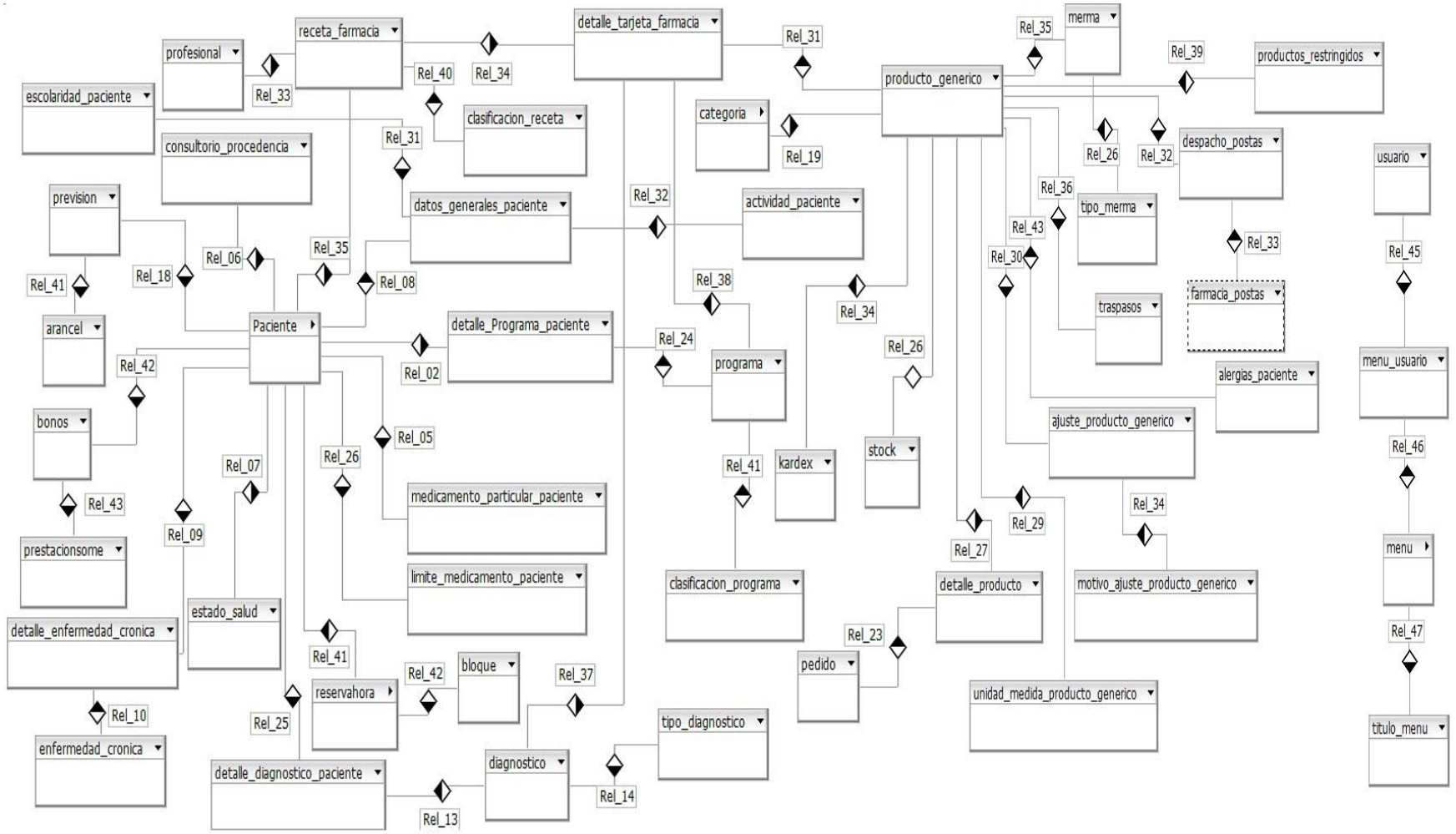



Figura 4: Modelo Entidad Relación, Sistema de Control de Farmacia

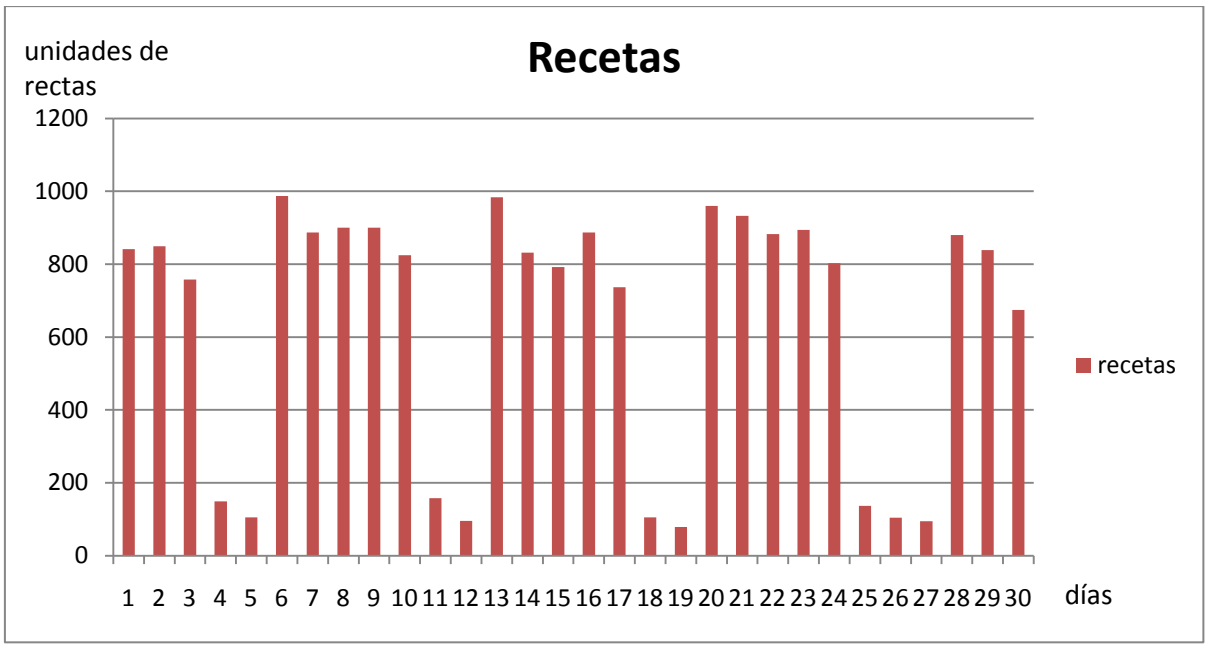



Figura 5: Recetas Entregadas Diariamente

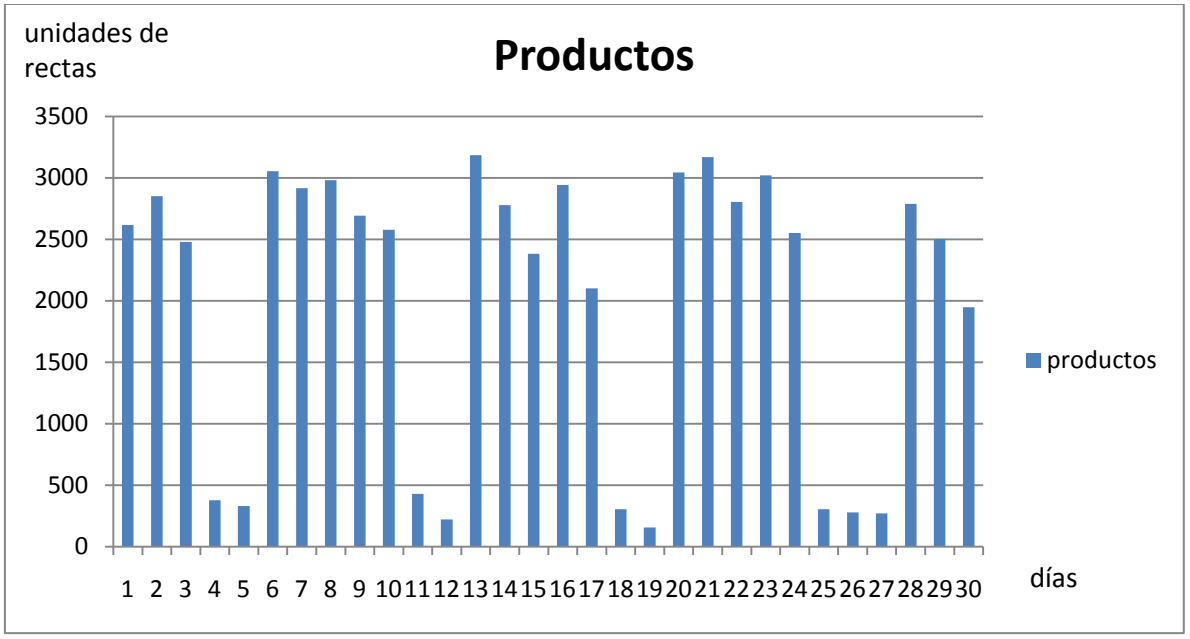



Figura 6: Productos Dispensados Diariamente

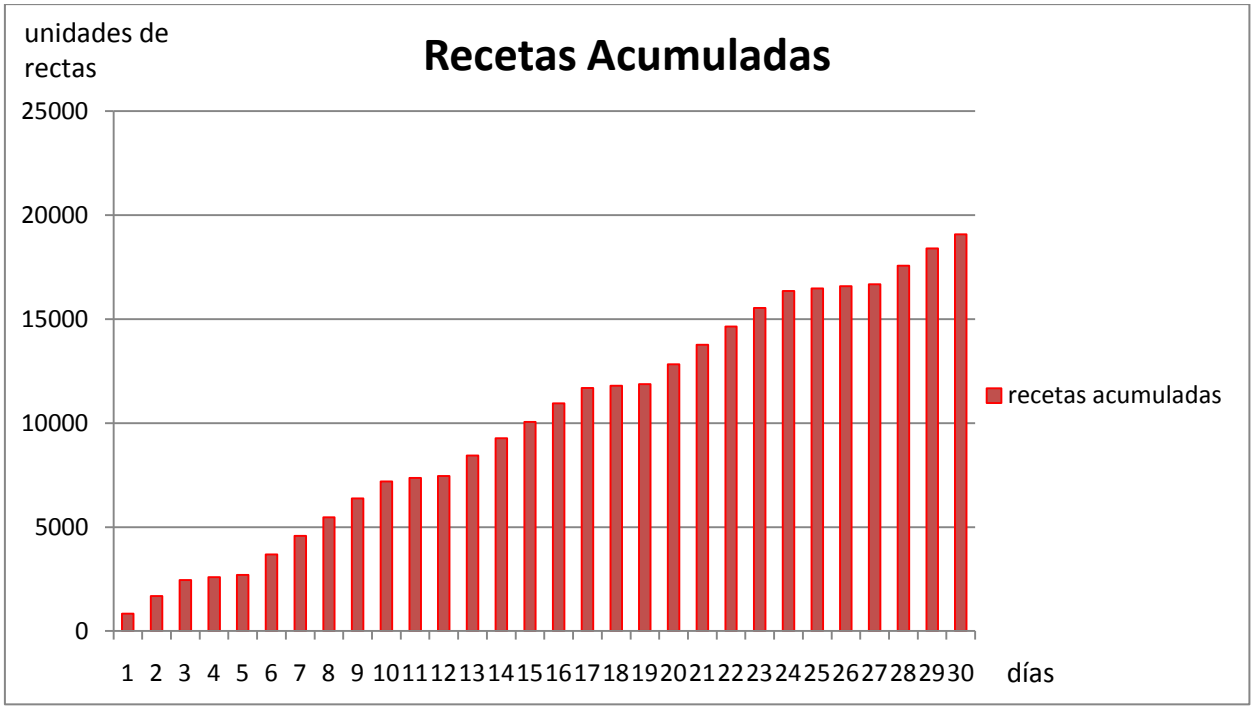



Figura 7: Recetas Entregadas Acumuladas Durante un Mes.

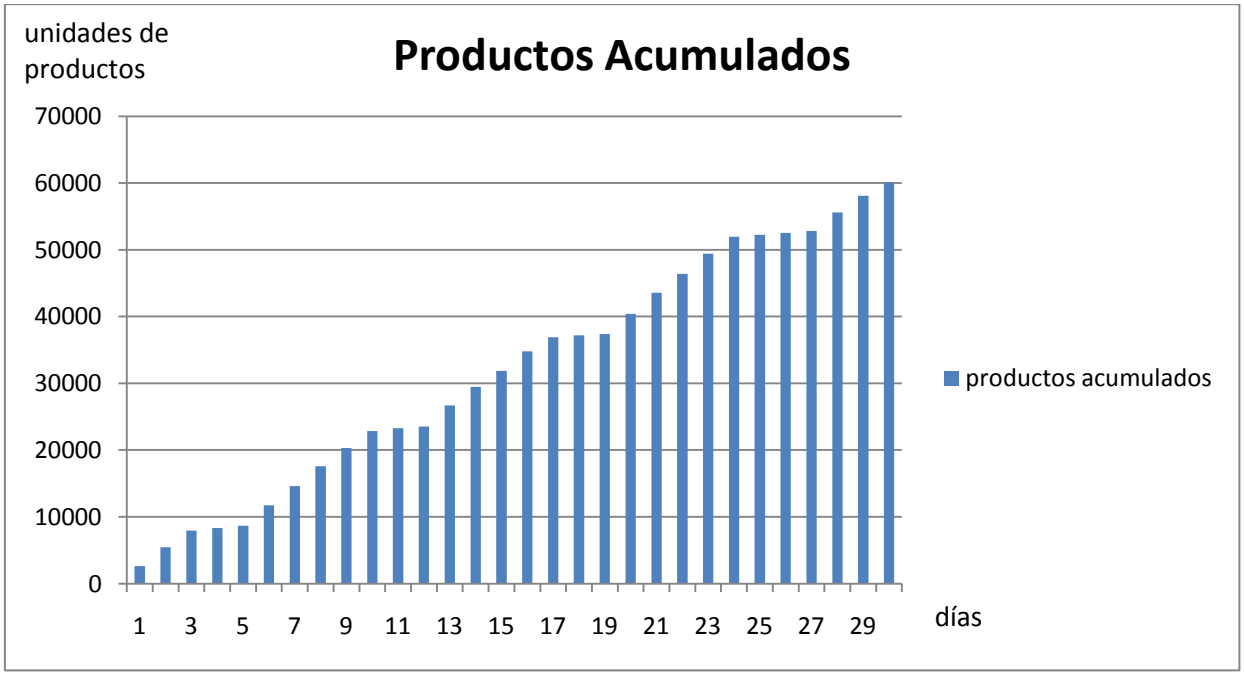



Figura 8: Productos Dispensados, Acumulados Durante un Mes.

# Capítulo III: Marco Teórico

## 3.1 Introducción

En este capítulo, se dan a conocer los recursos y herramientas utilizadas a lo largo de este estudio.

La intención de este capítulo no es proporcionar una guía de usuario completa para estos recursos. Más bien se presentan sus construcciones y conceptos fundamentales.

## 3.2 Conceptos Generales

En un sistema **de base de datos relacional** las consultas son expresadas en SQL, que es un lenguaje de alto nivel y declarativo, es decir, solo se especifica que información se quiere y no el cómo obtenerla (una sentencia no establece explícitamente un orden de ejecución) y es el SGBD quien selecciona la mejor estrategia de ejecución (Elmasri & Navathe, 2002).

El **procesamiento de consultas** son las actividades involucradas en la recuperación de datos este transforma una consulta SQL en una estrategia eficaz (existen muchas transformaciones equivalentes para una misma consulta), expresada en un lenguaje de bajo nivel y ejecuta dicha estrategia para recuperar los datos (Korth & Silberschatz, 1993).

La **optimización de consultas** es la elección de una estrategia de ejecución eficaz para procesar cada consulta sobre la base de datos para minimizar el uso de los recursos. En general no se garantiza que la estrategia seleccionada sea la más óptima pero sí que será una estrategia razonablemente eficiente (Tamer Ozsu & Valduriez, 1999).

La optimización de consultas suele combinar varias técnicas y dentro de las principales encontramos la **optimización heurística** que ordena las operaciones de las consultas para incrementar la eficiencia de uso (Tamer Ozsu & Valduriez, 1999).

En la **optimización de múltiples consultas** esta es una optimización que considera más de una consulta, que son ejecutadas simultáneamente y nos enfocaremos en las estrategias de ejecución de dichas consultas en base de datos relacionales, esto ayudara a la optimización del uso de la memoria principal y disminución de los accesos al disco del servidor.

## 3.3 El Lenguaje SQL

Este lenguaje, originalmente llamado Sequel, fue implementado como parte del proyecto R (de IBM) en los primeros años de la década de los setenta, desde entonces nunca ha dejado de evolucionar cambiando su nombre a SQL (Structured Query Language (Lenguaje de Consulta Estructurado)). Ahora se puede considerar como una de las razones más importantes del éxito de las base de datos relacionales en el mundo comercial, posesionándose como *el* lenguaje de base de datos relacional estándar (Korth & Silberschatz, 1993).

Dentro de este lenguaje podemos utilizar la herramienta llamada MySQL desarrollada por (Oracle and/or its affiliates, 2011). Que hace uso del lenguaje SQL y de todas sus propiedades, permitiéndonos crear tablas, eliminarlas, hacer consultas sobre ellas, etc.

### 3.3.1 MySQL

MySQL[2] es un sistema de gestión de base de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones siéndolo uno de los sistemas de gestión de base de datos Open Source más popular; que se desarrolla y distribuye con el apoyo de Oracle Corporation.

Existen varias APIs que permiten, a aplicaciones escritas en diversos lenguajes de programación, como por ejemplo java, acceder a las bases de datos MySQL.

### 3.3.2 Crear Tablas

La sentencia CREATE TABLE es la sentencia utilizada para crear una tabla y posee las siguientes características:

- CREATE TABLE “nom_tabla” crea una tabla con el nombre dado.
- Puede usar la palabra TEMPORARY al crear una tabla. Una tabla TEMPORARY es visible sólo para la conexión actual, y se borra automáticamente cuando la conexión se cierra. Esto significa que dos conexiones distintas pueden usar el mismo nombre de tabla temporal sin entrar en conflicto entre ellas ni con tablas no TEMPORARY con el mismo nombre. (La tabla existente se oculta hasta que se borra la tabla temporal.) En MySQL 5.0, se debe tener el permiso CREATE TEMPORARY TABLES para crear tablas temporales.
- En MySQL 5.0, se puede crear una tabla desde otra, añadiendo un comando SELECT al final del comando CREATE TABLE:
  “CREATE TABLE new_tbl SELECT * FROM orig_tbl”; (Oracle and/or its affiliates, 2011).

[2] Para mayor información visitar http://dev.mysql.com/

### 3.3.3 Insertar Datos

Con la sentencia INSERT es posible insertar datos en una tabla específica:

```
INSERT [LOW_PRIORITY | DELAYED | HIGH_PRIORITY] [IGNORE]
[INTO] tbl_name [(col_name,...)]
VALUES ({expr | DEFAULT},...),(...),...
[ ON DUPLICATE KEY UPDATE col_name=expr, ... ]
```

O:

```
INSERT [LOW_PRIORITY | HIGH_PRIORITY] [IGNORE]
[INTO] tbl_name [(col_name,...)]
SELECT ...
[ ON DUPLICATE KEY UPDATE col_name=expr, ... ]
```

Con INSERT ... SELECT, puede insertar rápidamente varios registros en una tabla desde una o varias tablas (Oracle and/or its affiliates, 2011).

Actualmente, no puede insertar en una tabla y seleccionar de la misma tabla en una sub-consulta.

### 3.3.4 Eliminar una Tabla

Con la sintaxis "DELETE" es posible eliminar (o borrar) una tabla:

- Sintaxis para una tabla:
  ```
  DELETE [LOW_PRIORITY] [QUICK] [IGNORE] FROM tbl_name
  [WHERE where_definition]
  [ORDER BY ...]
  [LIMIT row_count]
  ```
- Sintaxis para múltiples tablas:
  ```
  DELETE [LOW_PRIORITY] [QUICK] [IGNORE]
  tbl_name[.*] [, tbl_name[.*] ...]
  FROM table_references
  [WHERE where_definition]
  ```

DELETE borra los registros de tbl_name que satisfacen la condición dada por where_definition, y retorna el número de registros borrados.

Si realiza un comando DELETE sin cláusula WHERE se borran todos los registros. Una forma más rápida de hacerlo, cuando no quiere saber el número de registros borrados, se usa TRUNCATE TABLE.

Si especifica LOW_PRIORITY, la ejecución de DELETE se retarda hasta que no hay más clientes leyendo de la tabla (Oracle and/or its affiliates, 2011).

### 3.3.5 Bloquear Tablas

Con la sintaxis de LOCK TABLES y UNLOCK TABLES se puede bloquear y desbloquear una tabla para que un solo usuario haga uso de esta.

LOCK TABLES
tbl_name [AS alias] {READ [LOCAL] | [LOW_PRIORITY] WRITE}
[, tbl_name [AS alias] {READ [LOCAL] | [LOW_PRIORITY] WRITE}] ...
UNLOCK TABLES

Un bloqueo de tabla protege sólo contra lecturas inapropiadas o escrituras de otros clientes. El cliente que tenga el bloqueo, incluso un bloqueo de lectura, puede realizar operaciones a nivel de tabla tales como DROP TABLE.

Si un flujo obtiene un bloqueo READ en una tabla, ese flujo (y todos los otros) sólo pueden leer de la tabla. Si un flujo obtiene un bloqueo WRITE en una tabla, sólo el flujo con el bloqueo puede escribir en la tabla. El resto de flujos se bloquean hasta que se libera el bloqueo (Oracle and/or its affiliates, 2011).

### 3.3.6 Estructura básica de una Consulta SQL

La estructura básica de una expresión en SQL consta de tres clausulas:

- **SELECT,** se usa para listar los atributos que se desean en el resultado de una consulta.
- **FROM,** en esta clausula se deben listar todas las relaciones que se van ha examinar en la ejecución de la consulta.
- **WHERE,** consta de **condiciones**, que implican los atributos de las relaciones que aparecen en la clausula FROM, para distinguir (o seleccionar) que información es la que se requiere de esas relaciones. La condición de selección permite las comparaciones utilizando **=** (igualdad)**, <>** (desigualdad), **<** (menor que)**, =<** (menor igual que)**, >** (mayor que)**, = >** (mayor igual que)**;** combinando estos entre atributos o entre atributos y constantes. Pudiendo combinarse con los conectores **AND** (y) y **OR** (o)**.**

Una consulta típica en SQL tiene la siguiente forma:

**SELECT** listado de atributos
**FROM** listado de tablas
**WHERE** condición

Cada atributo en la clausula SELECT representa un atributo de una relación y cada tabla en la clausla FROM una relación de la base de datos. Si se omite la clausula **WHERE** se considera la condición de selección como verdadera. La lista de atributos puede sustituirse por un asterisco (*) para seleccionar todos los atributos de todas las relaciones que aparecen en la clausula **FROM**. Además una consulta se pude combinar con otra consulta utilizando las operaciones de conjunto **UNION** (unión), **INTERSECT** (intersección) y **MINUS** (diferencia) o utilizando una sub-consulta dentro de una consulta típica.

SQL permite el uso de funciones en grupos de tuplas usando la clausula **GROUP BY.** El atributo o atributos dados en la clausula **GROUP BY** se usan para formar grupos con tuplas del mismo valor (de los atributos en **GROUP BY**) en cada grupo. Las funciones que incluye SQL sirven para calcular:

- Promedio: **AVG**
- Mínimo: **MIN**
- Máximo: **MAX**
- Total: **SUM**
- Contar: **COUNT**

Estas son funciones de agregación ya que operan sobre grupos de tuplas y el resultado que entregan es un valor único.

Por lo general las consultas en SQL se descomponen en bloques de consultas, que constituyen las unidades básicas que pueden traducirse a operadores algebraicos para ser optimizadas. Un bloque de consulta contiene una única expresión SELECT-FROM-WHERE, así como las clausulas GROUP BY y HAVING si estas son partes del bloque. Por lo tanto, las consultas anidadas dentro de una consulta son identificadas como bloques de consultas distintos.

**Ejemplo 1 (parte 1): Obtención de Códigos de los Productos**

Consideremos las siguientes dos relaciones:
(1)**Receta_farmacia(**Folio,Fecha,Run_profesional,Run_paciente,Pendiente)
(2)**Detalle_tarjeta_farmacia(**Folio_receta_farmacia,Fecha_entrega,Código_ptoducto, Procedencia) de donde se solicita: “obtener el código de todos los productos prescritos del paciente con el Rut 9.078.656-3 ”. Para lograr obtener este resultado se puede realizar la siguiente consulta en SQL:

***SELECT*** *codigo_producto*
***FROM*** *receta_farmacia, detalle_tarjeta_farmacia*
***WHERE*** *folio=folio_receta_farmacia* ***AND*** *receta_farmacia.run_paciente* ***=*** *'9078656-3'*

## 3.4 Álgebra Relacional

El álgebra relacional es un lenguaje procedimental. Por lo tanto cada expresión del álgebra relacional representa una secuencia determinada de operaciones, se utilizarán paréntesis redundantes para indicar de forma no ambigua el orden de evaluación de las operaciones. El álgebra relacional es utilizado para llevar una consulta de alto nivel a una forma normal conjuntiva (o disyuntiva en algunos casos). Esta expresión en el álgebra relacional es equivalente con la consulta original en SQL, a la cual se puede llegar utilizando las reglas de transformación de expresiones (Elmasri & Navathe, 2002).

Dentro del álgebra relacional existen operaciones fundamentales: *seleccionar, proyectar, producto cartesiano, renombrar, unión y diferencia de conjuntos.* Además existen otras operaciones como la: *intersección de conjuntos, producto natural, división y asignación.*

La operación **seleccionar**, selecciona las tuplas que satisfacen un predicado dado, es denotado por la letra griega sigma (σ) y es equivalente a la clausula **WHERE** del lenguaje SQL. El predicado de selección permite las comparaciones utilizando ≠, =, <, ≤, >, ≥. Pudiendo combinarse con los conectores y (∧) y o (∨). La **proyección** se indica por la letra griega pi (π) y es el equivalente de la clausula **SELECT.** Este nos permite listar los atributos que queremos que aparezcan en el resultado, omitiendo el resto de las columnas. Estas operaciones son unarias, es decir, operan sobre una única relación.

El **producto cartesiano**, nos permite extraer información de dos relaciones y es denotada por una cruz (×) (por ejemplo r1 × r2), y entrega una relación, cuyo esquema corresponde a una combinación de todas las tuplas de r1 con cada una de las tuplas de r2, y sus atributos corresponden a los de r1 seguidos por los de r2. Suponiendo que $r_1$ posee $n_1$ tuplas y $r_2$ posee $n_2$ tuplas, entonces la relación resultante posee $n_1 * n_2$ tuplas.

Cuando se necesita la **unión** de dos consultas para entregar un resultado, se debe utilizar el operador binario de unión denotado por la teoría de conjuntos por ∪, pero para que una operación de unión sea válida se exigen dos condiciones: (1) ambas relaciones deben tener el mismo número de atributos. (2) Los dominios del atributo iésimo de una relación deben ser los mismos del atributo iésimo de la otra relación.

La operación de **diferencia de conjuntos**, representada por –, nos permite encontrar tuplas que estén en una relación pero no en otra. La expresión r – s (siendo r y s dos relaciones), da como resultado una relación que contiene las tuplas contenidas en r pero no en s.

La **intersección de conjuntos** denotada por: r1 ∩ r2; corresponde al conjunto de todas las tuplas que están en r1 y en r2, siendo r1 y r2 uniones compatibles.

La **unión natural** (NATURAL JOIN) en el álgebra relacional es la que permite reconstruir las tablas originales previas al proceso de normalización. Consiste en combinar las proyección, selección y producto cartesiano en una sola operación, donde la condición θ es la igualdad Clave Primaria = Clave Externa (o Foránea), y la proyección elimina la columna duplicada (clave externa). Expresada en las operaciones básicas, queda:

$R \bowtie S = \pi_{A1,A2,\ldots,An}(\sigma_{\theta} (S \times R))$

Una reunión theta ( θ-Join) de dos relaciones es equivalente a:

$R \bowtie_{\theta} S = (\sigma_{\theta} (S \times R))$ donde la condición θ es libre y si la condición θ es una igualdad se denomina EquiJoin.

**División,** supongamos que tenemos dos relaciones A(x, y) y B(y) donde el dominio de y en A y B, es el mismo. El operador división A / B retorna todos los distintos valores de x tales que para todo valor y en B existe una tupla en A.

Una expresión general en el álgebra relacional se construye a partir de sub-expresiones. Sean A y E expresiones del álgebra relacional. Entonces las siguientes son todas expresiones del álgebra relacional:

- $A \cup E$
- $A - E$
- $A \times E$
- $\sigma_p(A)$, donde p es un predicado de atributos de A
- $\pi_s(A)$, donde s es una lista que consta de algunos de los atributos de A

Se debe conocer estos operadores y las reglas de transformación de expresiones aritméticas pues aparecen en las consultas. Además se debe saber aplicar las reglas generales de los operadores condicionales de comparación (<,>,…) y lógicos (AND, OR,…) para así llevar una expresión de alto nivel (lenguaje SQL) a una expresión de bajo nivel (lenguaje del álgebra relacional).

**Ejemplo 1 (parte 2): Obtención de Códigos de los Productos**

Consideremos la consulta, que se encuentra en un lenguaje de alto nivel, del Ejemplo en la Sección 3.3.7 que obtenía los productos prescritos de un determinado paciente:

***SELECT*** *codigo_producto*

***FROM*** *receta_farmacia, detalle_tarjeta_farmacia*
***WHERE*** *folio=folio_receta_farmacia*
***AND*** *receta_farmacia.run_paciente = '9078656-3'*

La consulta equivalente en el álgebra relacional correspondería a la siguiente expresión:

$$\Pi \text{ codigo\_producto } (\sigma \text{ run\_paciente} = 9078656\text{-}3 \wedge \text{receta\_farmacia.folio} = \text{folio\_receta\_farmacia } (\text{receta\_farmacia} \times \text{detalle\_tarjeta\_farmacia}))$$

## 3.5 Árboles de Consulta y Grafos de Consulta

Un árbol o grafo de consulta se utiliza para representar una consulta, generalmente una consulta ya expresada en el álgebra relacional. Esta representación es equivalente con su consulta original.

Un árbol de consulta es una estructura de datos que corresponde a una expresión del álgebra relacional. Representa las relaciones de entrada de una consulta como nodos hoja del árbol y las operaciones del álgebra relacional como nodos internos. Una ejecución del árbol de consulta consiste en la ejecución de una operación de un nodo interno siempre que sus operandos estén disponibles, sustituyendo después ese nodo interno por la relación que resulte de ejecutar la operación. La ejecución termina cuando se ejecuta el nodo raíz y produce la relación resultado para la consulta. A continuación se describe un árbol de consulta mediante un ejemplo presentado por (Elmasri & Navathe, 2002).

**Ejemplo 1 (parte 3) Obtención de Códigos de los Productos**

La Figura 9 muestra un árbol de consulta de expresión del álgebra relacional obtenida del Ejemplo 1 (parte 2) de la Sección 3.4.1 mencionada anteriormente.

Cuando se ejecuta este árbol debe comenzar su ejecución por las hojas (desde abajo hacia arriba) existen ocasiones en que las tuplas resultantes del nodo A son necesarias para otro nodo B por lo que se debe ejecutar primero el nodo A.

Como se puede ver, el árbol de consulta representa un orden específico de operaciones para ejecutar una consulta. Una representación más neutral de la consulta es la notación de grafo de consulta.

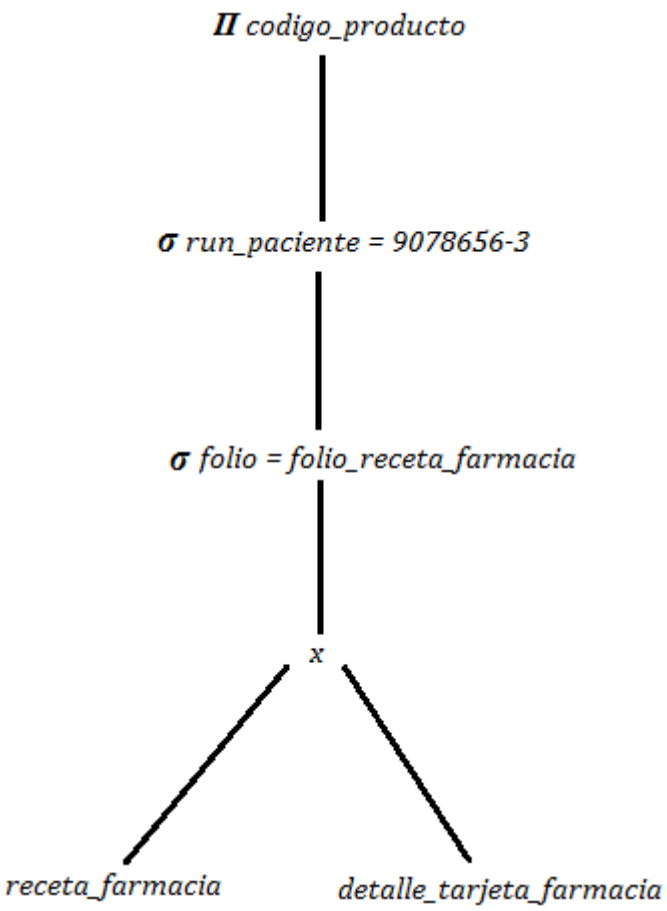



Figura 9: Árbol de consulta

La Figura 10 muestra el grafo de consulta para la misma consulta anterior. Las relaciones que se establecen en la consulta se representan mediante nodos de relación, que se representan como círculos únicos.

Los valores constantes, que son característicos de las condiciones de selección de consultas, se representan mediante arcos (o aristas) que afectas a una única relación (o nodo). Las condiciones de selección/reunión se representan mediante aristas del grafo que conectan dos relaciones como, se muestra en la Figura 12 (b). Finalmente los atributos que se han de obtener de cada relación aparecen entre paréntesis cuadrados sobre cada relación.

La representación del grafo de consulta no indica en qué orden se deben realizar las operaciones en primer lugar. Aunque algunas técnicas de optimización estaban basadas en grafos de consultas, ahora se ha aceptado de forma generalizada que los arboles de consulta son preferibles porque, en la práctica, el optimizador de consultas necesita mostrar el orden de operación para la ejecución de la consulta.

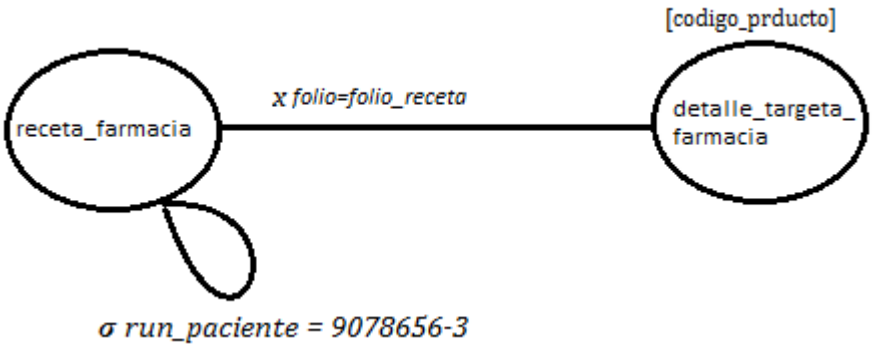



Figura 10: Grafo de Consulta.

## 3.6 Herramientas

### 3.6.1 JasperReports

JasperReports es una herramienta de creación de informes Java que tiene la habilidad de entregar contenido enriquecido al monitor, a la impresora o a archivos PDF, HTML, XLS, CSV y XML.

Está escrito completamente en Java y puede ser usado en gran variedad de aplicaciones de Java, incluyendo J2EE o aplicaciones web, para generar contenido dinámico. Su propósito principal es ayudar a crear documentos de tipo páginas, preparados para imprimir en una forma simple y flexible. Ésta herramienta es utilizada comúnmente con iReport (ITEM, 2010).

### 3.6.2 MyEclipse

MyEclipse[3] es un entorno de desarrollo que está disponible en el mercado de Java EE y AJAX IDE, creado y mantenido por la empresa Genuitec, miembro fundador de la Fundación Eclipse.

MyEclipse se basa en la plataforma Eclipse e integra soluciones de código abierto y propietario en el entorno de desarrollo.

Para el desarrollo del sistema de control de farmacia se utilizó MyEclipse 8.5 este es un entorno de desarrollo integrado completo y asequible (IDE) basado en la plataforma de Eclipse centrándose principalmente en la mejora de la productividad de los desarrolladores mediante la simplificación del ciclo de vida del desarrollo en la entrega de UML, Web, J2EE, XML, JSP, JSF, Struts, Hibernate y las aplicaciones de base de datos. Esta versión de MyEclipse se centra principalmente en que permite la colaboración entre los desarrolladores del equipo y ofrece cientos de nuevos desarrollos y mejora de código, pruebas y características de implementación.

### 3.6.3 SQL-Front 5.1

SQL-Front[4] 5.1 Es una herramienta stand-alone que simplifica la creación y el cambio de cualquier objeto de base de datos, añadir y modificar cualquier registro de base de datos y un montón de pasos además el desarrollo de aplicaciones con acceso a bases de datos en todos los idiomas (Oracle and/or its affiliates, 2011).

### 3.6.4 Ireport

IReport[5] es un diseñador de informes visual, poderoso, intuitivo y fácil de usar para JasperReports escrito en Java. Este instrumento permite que los usuarios corrijan

[3] Para mayor información sobre esta aplicación visitar: http://www.myeclipseide.com/
[4] http://www.mysqlfront.de/wp/
[5] http://jasperforge.org/projects/ireport

visualmente informes complejos con cartas, imágenes, sub-informes, etc. iReport está además integrado con JFreeChart, una de la biblioteca gráficas OpenSource más difundida para Java. Los datos para imprimir pueden ser recuperados por varios caminos, incluso múltiples uniones JDBC, TableModels, JavaBeans, XML, entre otros (Herrera, 2005) **.**

Ireport 3.7.3: es la versión utilizada en este proyecto con el cual se pueden crear diseños muy sofisticados que contienen gráficos, imágenes, sub-informes, referencias cruzadas y mucho más. Se puede acceder a sus datos a través de JDBC, TableModels, JavaBeans, XML, Hibernate, CSV y fuentes personalizadas y pudiendo publicar los informes en formato PDF, RTF, XML, CSV, HTML, XHTML, texto, DOCX o OpenOffice.

## 3.7 Recursos

### 3.7.1 Servidor de aplicaciones (*Application Server*)

Si bien el sistema de control de farmacia cumple con el modelo de aplicación de tres capas (ver Figura 11) físicamente la aplicación y la base de datos residen en el mismo servidor, este servidor es un servidor de aplicación "HP ProLiant ML150 G6[6]", en el anexo B: Tipos de servidores, se muestra un análisis de los diferentes tipos de servidores más utilizados; las características que posee este servidor se encuentran en la Tabla 2. Este servidor esta diseñado para satisfacer sus necesidades de TI de hoy en día, mientras que proporciona nuevas capacidades para el futuro. El HP ProLiant ML150 G6 es muy adecuado para las pequeñas y medianas empresas, así como de computación intensiva entornos de clúster de servidor que requieren un rendimiento confiable. Equipado con tecnologías estándar del sector, el ML150 G6 proporciona la potencia necesaria para hacer frente a las exigentes aplicaciones empresariales tales como:

- Archivo e impresión.
- Los pequeños y medianos servidores de correo electrónico.
- Pequeñas aplicaciones verticales o bases de datos.
- Acceso a Internet compartido e infraestructura LAN

[6] http://h10010.www1.hp.com/wwpc/es/es/sm/WF06a/15351-15351-241434-3328424-3328424-3884323.html?dnr=1&jumpid=in_r2515_es/es/smb/psg/psc404redir-ot-xx-xx-/chev/

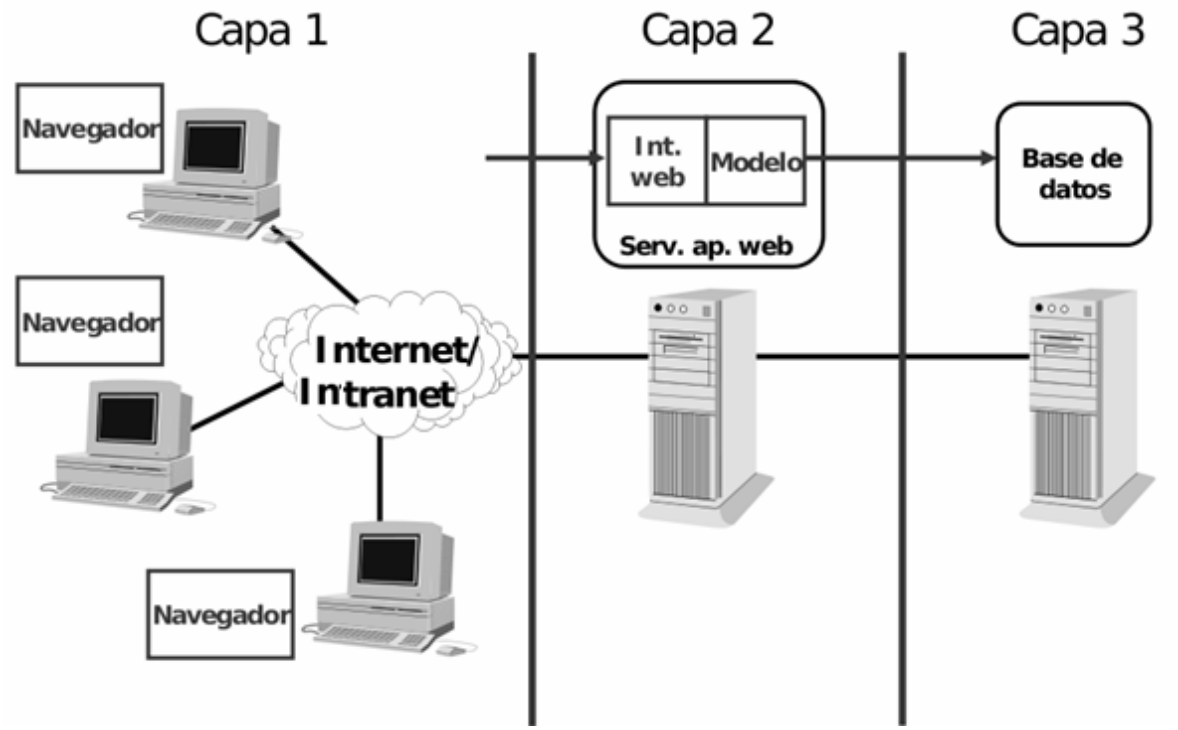



Figura 11: Modelo de Aplicación de Tres Capas

| | |
|---|---|
| Procesador: | **Intel® Xeon® E5504 (4 núcleos, 2,00 GHz, 4 MB L3, 80W).** |
| Memoria estándar: | 2 GB (máxima 48 GB). |
| Memoria actual: | 6 GB. |
| Sistema operativo: | Windows Server 6 Enterprise |
| Ranuras de memoria: | 12 ranuras DIMM |
| Tipo de memoria | PC3-10600E (UDIMM) |

Tabla 2: Características del Servidor HP Proliant ML150.

Este servidor además de mantener el servicio del control de farmacia también se encarga de otras aplicaciones como:

- Agenda de los profesionales.
- Reserva de horas.
- Recaudación.
- Atención de urgencia.
- Lista de espera.

Se estima que todos los sistemas y aplicaciones hacen igual uso del servidor.

# Capítulo IV: Problemas Detectados en el Sistema de Control de Farmacia.

## 4.1 Introducción

La necesidad de este estudio nace ya que el consultorio Violeta Parra desea mejorar los tiempos de respuestas y optimizar el sistema de "control de farmacia". Específicamente el consultorio identifica problemas presentes en la obtención de dos reportes del módulo de estadísticas, que perteneciente al sistema de "control de farmacia". En este capítulo se realiza un análisis de los problemas que presenta el sistema de farmacia, dando así a conocer la propuesta de solución y casos de estudios que se consideran en este proyecto.

## 4.2 Problemas Detectados

A continuación se presenta el análisis de los problemas de los reportes ya mencionados y otros problemas en el sistema de control de farmacia detectados. Estos problemas se detectaron mediantes una revisión de los reportes mencionados, las consultas que los generan y la base de datos existente.

### 4.2.1 Reporte Estadística Farmacia

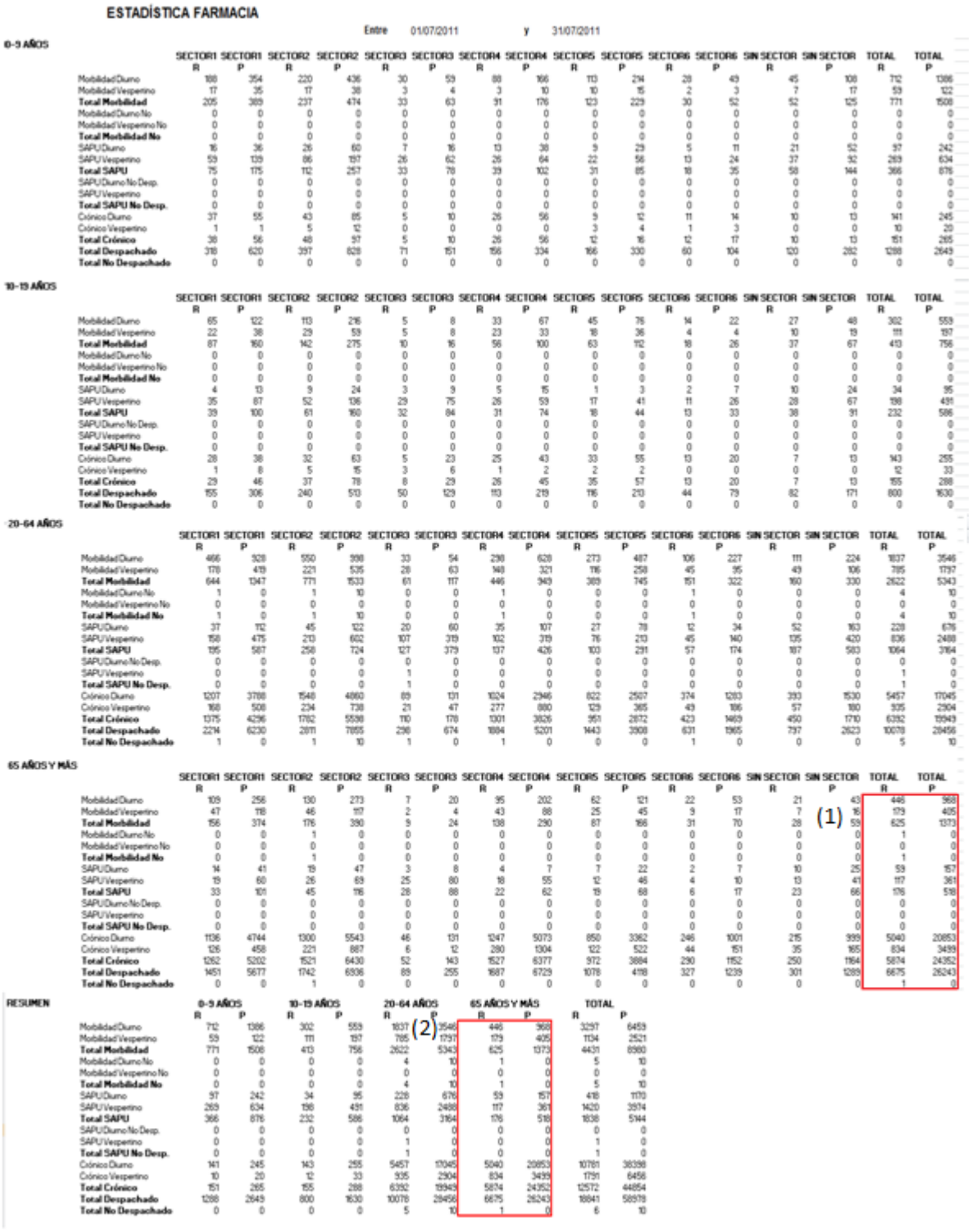
Al momento de ser solicitado este reporte tarda aproximadamente 8 minutos con 30 segundos, lo que para el Consultorio es un tiempo excesivo y desearía poder reducirlo a 3 minutos aproximadamente.

Este reporte entrega una estadística de todas las recetas y prescripciones dispensadas dentro de un rango de fechas establecido por el usuario, las recetas y prescripciones son primero desglosadas por la procedencia del paciente (crónico, morbilidad y SAPU) indicando si fueron despachadas o no, y si fueron en un horario diurno o vespertino, luego se clasifican por grupo etario y por sector (recordar la Tabla 2: Población inscrita, de la sección 2.2.4 donde se desglosa por sectores).

Como se observa en la Figura 12, el reporte primero está dividido en 4 secciones por grupo etario (rango de edades) y cada una de ellas por los sectores de donde pertenece cada paciente y al final se presenta un resumen (donde no se consideran los sectores). Además en el Anexo C se presenta de una forma más clara este reporte.

Las columnas divididas por grupo etario en la sección del resumen deben coincidir con los totales de las secciones anteriores. En la parte inferior de la figura 12 se destacan con cuadros rojos: (1) los totales de las recetas y prescripciones para pacientes pertenecientes al grupo etario mayores de 65 años y (2) el grupo de pacientes mayores de 65 años en el resumen del reporte, estos datos se generan con la misma consulta por lo que siempre coincidirán, es decir, se está ejecuta dos veces la

misma consulta para cada grupo etario. Esto ocurre de la misma forma con el resto de los grupos etarios.



ESTADÍSTICA FARMACIA

Entre 01/07/2011 y 31/07/2011

0-9 AÑOS

| | SECTOR1 R | SECTOR1 P | SECTOR2 R | SECTOR2 P | SECTOR3 R | SECTOR3 P | SECTOR4 R | SECTOR4 P | SECTOR5 R | SECTOR5 P | SECTOR6 R | SECTOR6 P | SIN SECTOR R | SIN SECTOR P | TOTAL R | TOTAL P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Morbilidad Diurno | 168 | 354 | 220 | 436 | 30 | 59 | 88 | 166 | 113 | 214 | 28 | 49 | 45 | 108 | 712 | 1386 |
| Morbilidad Vespertino | 17 | 35 | 17 | 38 | 3 | 4 | 3 | 10 | 10 | 15 | 2 | 3 | 7 | 17 | 59 | 122 |
| **Total Morbilidad** | 205 | 389 | 237 | 474 | 33 | 63 | 91 | 176 | 123 | 229 | 30 | 52 | 52 | 125 | 771 | 1508 |
| Morbilidad Diurno No | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Morbilidad Vespertino No | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Total Morbilidad No** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| SAPU Diurno | 16 | 36 | 26 | 60 | 7 | 16 | 13 | 38 | 9 | 29 | 5 | 11 | 21 | 52 | 97 | 242 |
| SAPU Vespertino | 59 | 139 | 86 | 197 | 26 | 62 | 26 | 64 | 22 | 56 | 13 | 24 | 37 | 92 | 269 | 634 |
| **Total SAPU** | 75 | 175 | 112 | 257 | 33 | 78 | 39 | 102 | 31 | 85 | 18 | 35 | 58 | 144 | 366 | 876 |
| SAPU Diurno No Desp. | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| SAPU Vespertino | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Total SAPU No Desp.** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Crónico Diurno | 37 | 55 | 43 | 85 | 5 | 10 | 26 | 56 | 9 | 12 | 11 | 14 | 10 | 13 | 141 | 245 |
| Crónico Vespertino | 1 | 1 | 5 | 12 | 0 | 0 | 0 | 0 | 3 | 4 | 1 | 3 | 0 | 0 | 10 | 20 |
| **Total Crónico** | 38 | 56 | 48 | 97 | 5 | 10 | 26 | 56 | 12 | 16 | 12 | 17 | 10 | 13 | 151 | 265 |
| **Total Despachado** | 318 | 620 | 397 | 828 | 71 | 151 | 156 | 334 | 166 | 330 | 60 | 104 | 120 | 282 | 1288 | 2649 |
| **Total No Despachado** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

10-19 AÑOS

| | SECTOR1 R | SECTOR1 P | SECTOR2 R | SECTOR2 P | SECTOR3 R | SECTOR3 P | SECTOR4 R | SECTOR4 P | SECTOR5 R | SECTOR5 P | SECTOR6 R | SECTOR6 P | SIN SECTOR R | SIN SECTOR P | TOTAL R | TOTAL P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Morbilidad Diurno | 65 | 122 | 113 | 216 | 5 | 8 | 33 | 67 | 45 | 76 | 14 | 22 | 27 | 48 | 392 | 559 |
| Morbilidad Vespertino | 22 | 38 | 29 | 59 | 5 | 8 | 23 | 33 | 18 | 36 | 4 | 4 | 10 | 19 | 111 | 197 |
| **Total Morbilidad** | 87 | 160 | 142 | 275 | 10 | 16 | 56 | 100 | 63 | 112 | 18 | 26 | 37 | 67 | 413 | 756 |
| Morbilidad Diurno No | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Morbilidad Vespertino No | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Total Morbilidad No** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| SAPU Diurno | 4 | 13 | 9 | 24 | 3 | 9 | 5 | 15 | 1 | 3 | 2 | 7 | 10 | 24 | 34 | 95 |
| SAPU Vespertino | 35 | 87 | 52 | 136 | 29 | 75 | 26 | 59 | 17 | 41 | 11 | 26 | 28 | 67 | 198 | 491 |
| **Total SAPU** | 39 | 100 | 61 | 160 | 32 | 84 | 31 | 74 | 18 | 44 | 13 | 33 | 38 | 91 | 232 | 586 |
| SAPU Diurno No Desp. | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| SAPU Vespertino | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Total SAPU No Desp.** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Crónico Diurno | 28 | 38 | 32 | 63 | 5 | 23 | 25 | 43 | 33 | 55 | 13 | 20 | 7 | 13 | 143 | 255 |
| Crónico Vespertino | 1 | 8 | 5 | 15 | 3 | 6 | 1 | 2 | 2 | 2 | 0 | 0 | 0 | 0 | 12 | 33 |
| **Total Crónico** | 29 | 46 | 37 | 78 | 8 | 29 | 26 | 45 | 35 | 57 | 13 | 20 | 7 | 13 | 155 | 288 |
| **Total Despachado** | 155 | 306 | 240 | 513 | 50 | 129 | 113 | 219 | 116 | 213 | 44 | 79 | 82 | 171 | 800 | 1630 |
| **Total No Despachado** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

20-64 AÑOS

| | SECTOR1 R | SECTOR1 P | SECTOR2 R | SECTOR2 P | SECTOR3 R | SECTOR3 P | SECTOR4 R | SECTOR4 P | SECTOR5 R | SECTOR5 P | SECTOR6 R | SECTOR6 P | SIN SECTOR R | SIN SECTOR P | TOTAL R | TOTAL P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Morbilidad Diurno | 466 | 928 | 550 | 998 | 33 | 54 | 298 | 628 | 273 | 487 | 106 | 227 | 111 | 224 | 1837 | 3546 |
| Morbilidad Vespertino | 178 | 419 | 221 | 535 | 28 | 63 | 148 | 321 | 116 | 258 | 45 | 95 | 43 | 106 | 785 | 1737 |
| **Total Morbilidad** | 644 | 1347 | 771 | 1533 | 61 | 117 | 446 | 949 | 389 | 745 | 151 | 322 | 160 | 330 | 2622 | 5343 |
| Morbilidad Diurno No | 1 | 0 | 1 | 10 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 4 | 10 |
| Morbilidad Vespertino No | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Total Morbilidad No** | 1 | 0 | 1 | 10 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 4 | 10 |
| SAPU Diurno | 37 | 112 | 45 | 122 | 20 | 60 | 35 | 107 | 27 | 78 | 12 | 34 | 52 | 163 | 228 | 676 |
| SAPU Vespertino | 158 | 475 | 213 | 602 | 107 | 319 | 102 | 319 | 76 | 213 | 45 | 140 | 135 | 420 | 836 | 2488 |
| **Total SAPU** | 195 | 587 | 258 | 724 | 127 | 379 | 137 | 426 | 103 | 291 | 57 | 174 | 187 | 583 | 1064 | 3164 |
| SAPU Diurno No Desp. | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| SAPU Vespertino | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| **Total SAPU No Desp.** | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| Crónico Diurno | 1207 | 3788 | 1548 | 4060 | 89 | 131 | 1024 | 2946 | 822 | 2507 | 374 | 1203 | 393 | 1530 | 5457 | 17045 |
| Crónico Vespertino | 168 | 508 | 234 | 738 | 21 | 47 | 277 | 880 | 129 | 365 | 43 | 186 | 57 | 180 | 935 | 2904 |
| **Total Crónico** | 1375 | 4296 | 1782 | 5598 | 110 | 178 | 1301 | 3826 | 951 | 2872 | 423 | 1469 | 450 | 1710 | 6392 | 19949 |
| **Total Despachado** | 2214 | 6230 | 2811 | 7855 | 298 | 674 | 1884 | 5201 | 1443 | 3908 | 631 | 1965 | 797 | 2623 | 10078 | 28456 |
| **Total No Despachado** | 1 | 0 | 1 | 10 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 5 | 10 |

65 AÑOS Y MÁS

| | SECTOR1 R | SECTOR1 P | SECTOR2 R | SECTOR2 P | SECTOR3 R | SECTOR3 P | SECTOR4 R | SECTOR4 P | SECTOR5 R | SECTOR5 P | SECTOR6 R | SECTOR6 P | SIN SECTOR R | SIN SECTOR P | TOTAL R | TOTAL P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Morbilidad Diurno | 109 | 256 | 130 | 273 | 7 | 20 | 95 | 202 | 62 | 121 | 22 | 53 | 21 | 43 | 448 | 968 |
| Morbilidad Vespertino | 47 | 118 | 46 | 117 | 2 | 4 | 43 | 88 | 25 | 45 | 9 | 17 | 7 | 16 | 179 | 405 |
| **Total Morbilidad** | 156 | 374 | 176 | 390 | 9 | 24 | 138 | 290 | 87 | 166 | 31 | 70 | 28 | 59 | 625 | 1373 |
| Morbilidad Diurno No | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| Morbilidad Vespertino No | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Total Morbilidad No** | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| SAPU Diurno | 14 | 41 | 19 | 47 | 3 | 8 | 4 | 7 | 7 | 22 | 2 | 7 | 10 | 25 | 59 | 157 |
| SAPU Vespertino | 19 | 60 | 26 | 69 | 25 | 80 | 18 | 55 | 12 | 46 | 4 | 10 | 13 | 41 | 117 | 361 |
| **Total SAPU** | 33 | 101 | 45 | 116 | 28 | 88 | 22 | 62 | 19 | 68 | 6 | 17 | 23 | 66 | 176 | 518 |
| SAPU Diurno No Desp. | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| SAPU Vespertino | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Total SAPU No Desp.** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Crónico Diurno | 1136 | 4744 | 1300 | 5543 | 46 | 131 | 1247 | 5073 | 850 | 3362 | 246 | 1001 | 215 | 999 | 5040 | 20853 |
| Crónico Vespertino | 126 | 458 | 221 | 887 | 6 | 12 | 280 | 1304 | 122 | 522 | 44 | 151 | 35 | 185 | 834 | 3439 |
| **Total Crónico** | 1262 | 5202 | 1521 | 6430 | 52 | 143 | 1527 | 6377 | 972 | 3884 | 290 | 1152 | 250 | 1184 | 5874 | 24352 |
| **Total Despachado** | 1451 | 5677 | 1742 | 6936 | 89 | 255 | 1887 | 6729 | 1078 | 4118 | 327 | 1239 | 301 | 1285 | 6675 | 26243 |
| **Total No Despachado** | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

(1)

RESUMEN

| | 0-9 AÑOS R | 0-9 AÑOS P | 10-19 AÑOS R | 10-19 AÑOS P | 20-64 AÑOS R | 20-64 AÑOS P | 65 AÑOS Y MÁS R | 65 AÑOS Y MÁS P | TOTAL R | TOTAL P |
|---|---|---|---|---|---|---|---|---|---|---|
| Morbilidad Diurno | 712 | 1386 | 392 | 559 | 1837 | 3546 | 448 | 968 | 3297 | 6459 |
| Morbilidad Vespertino | 59 | 122 | 111 | 197 | 785 | 1737 | 179 | 405 | 1134 | 2521 |
| **Total Morbilidad** | 771 | 1508 | 413 | 756 | 2622 | 5343 | 625 | 1373 | 4431 | 8980 |
| Morbilidad Diurno No | 0 | 0 | 0 | 0 | 4 | 10 | 1 | 0 | 5 | 10 |
| Morbilidad Vespertino No | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Total Morbilidad No** | 0 | 0 | 0 | 0 | 4 | 10 | 1 | 0 | 5 | 10 |
| SAPU Diurno | 97 | 242 | 34 | 95 | 228 | 676 | 59 | 157 | 418 | 1170 |
| SAPU Vespertino | 269 | 634 | 198 | 491 | 836 | 2488 | 117 | 361 | 1420 | 3974 |
| **Total SAPU** | 366 | 876 | 232 | 586 | 1064 | 3164 | 176 | 518 | 1838 | 5144 |
| SAPU Diurno No Desp. | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| SAPU Vespertino | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| **Total SAPU No Desp.** | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| Crónico Diurno | 141 | 245 | 143 | 255 | 5457 | 17045 | 5040 | 20853 | 10781 | 38398 |
| Crónico Vespertino | 10 | 20 | 12 | 33 | 935 | 2904 | 834 | 3439 | 1791 | 6456 |
| **Total Crónico** | 151 | 265 | 155 | 288 | 6392 | 19949 | 5874 | 24352 | 12572 | 44854 |
| **Total Despachado** | 1288 | 2649 | 800 | 1630 | 10078 | 28456 | 6675 | 26243 | 18841 | 58979 |
| **Total No Despachado** | 0 | 0 | 0 | 0 | 5 | 10 | 1 | 0 | 6 | 10 |

(2)

Figura 12: Reporte Estadística Farmacia.

**ESTADÍSTICA FARMACIA**

**0-9 AÑOS**

Ver figura 14

| | SECTOR1 R | SECTOR1 P |
|---|---|---|
| Morbilidad Diurno | 188 | 354 |
| Morbilidad Vespertino | 17 | 35 |
| **Total Morbilidad** | 205 | 389 |
| Morbilidad Diurno No | 0 | 0 |
| Morbilidad Vespertino No | 0 | 0 |
| **Total Morbilidad No** | 0 | 0 |
| SAPU Diurno | 16 | 36 |
| SAPU Vespertino | 59 | 139 |
| **Total SAPU** | 75 | 175 |
| SAPU Diurno No Desp. | 0 | 0 |
| SAPU Vespertino | 0 | 0 |
| **Total SAPU No Desp.** | 0 | 0 |
| Crónico Diurno | 37 | 55 |
| Crónico Vespertino | 1 | 1 |
| **Total Crónico** | 38 | 56 |
| **Total Despachado** | 318 | 620 |
| **Total No Despachado** | 0 | 0 |

Figura 13: Primera Sección Reporte Estadística Farmacia.

La Figura 13 es una ampliación de la primera sección donde se muestra el total de recetas y prescripciones por pacientes del sector 1 pertenecientes al grupo etario de 0 a 9 años, según su procedencia (morbilidad, SAPU y crónico) indicando si fueron despachadas en horario de atención diurno o vespertino y si fueron despachadas o no. Para calcular estas dos columnas (recetas y prescripciones) con sus respectivas filas (morbilidad diurno, morbilidad vespertino, morbilidad diurno no despechadas, …, crónico diurno, crónico vespertino), se ejecuta una consulta que contiene 24 sub-consultas (a pesar de que los valores que se muestran son solo 20), es decir, para calcular todos los valores del reporte completo se ejecutan 37 consultas con 888 sub-consultas en total (37 consultas x 24 sub-consultas).

En la Figura 14 (a) se muestra la sub-consulta que calcula las recetas despachadas de los pacientes de procedencia morbilidad, despachadas en horario vespertino del sector 1, pertenecientes al grupo etario de 0 a 9 años (Figura 13, valor en el recuadro en rojo). Como se muestra en las cuadros marcados con rojo de la consulta en Figura 14 (a) se observa como dentro de la consulta se repinten predicados, esto sucede en cada una de las consultas y sub-consultas ejecutadas de todo el reporte. Además en este reporte, al igual que en el reporte de monitoreo diario, al momento de calcular el número de recetas y prescripciones, despachadas y no despachadas cometen un error, este error fue producido por una mala captación de requisitos (ya que la formula actual, para contar (o calcular) las recetas y prescripciones, que se utiliza se debe aplicar solo a los pacientes de procedencia

crónica), en otras palabras, para los pacientes crónicos se debe considerar para cuantos meses de tratamiento se han despachado los productos. Por ejemplo si se entregan productos para 60 días se debe considerar como dos recetas. Además los pacientes crónicos están constantemente retirando remedios, por lo que se tratan de forma distinta, en cambio para un paciente de morbilidad esta receta contaría como una sola receta; esto provoca cálculos innecesarios y datos erróneos. Como podemos ver en la consulta de la Figura número 14 (del reporte de estadísticas) el cálculo para obtener las fechas de próximas entregas y la suma de estas, es innecesario y solo bastaría con contar cada receta por ejemplo con un “count(receta_farmacia.folio)”.


Figura 14 (a): Sub-consulta Recetas Despachadas por Pacientes de Morbilidad en Horario Vespertino Pertenecientes al Sector 1 y al Grupo Etario 1.

En la figura 14 (a) se detalla una de las múltiples sub-consultas necesarias para generar este reporte. Ya que dentro de las diferentes sub-consulta existen pequeñas diferencias a partir de esta sub-consulta se pueden generar las distintas sub-consultas pertenecientes a esta sección del reporte por ejemplo:

- Para obtener la sub-consulta que obtiene las recetas de procedencia morbilidad diurno vasta con cambiar los signos de operación al momento de preguntar por la hora de entrega, de la siguiente manera:

```
cast(receta_farmacia.fecha as time)>='16:00:00'
→ and cast(receta_farmacia.fecha as time)<'16:00:00'
```

  En todos los casos correspondientes dentro de esta sub-consulta.
- Para obtener la sub-consulta que obtiene las recetas de procedencia crónica o SAPU vasta con cambiar la procedencia de la siguiente manera:

```
and detalle_tarjeta_farmacia.procedencia='Morbilidad'
→and detalle_tarjeta_farmacia.procedencia='Crónico'
```

  Pero SAPU atiende todos los días en un mismo horario "hasta las 18:00 horas" antes de las 18:00 horas se considera diurno, después de esta hora vespertino.
- Para obtener las prescripciones envés de obtener las recetas se debe modificar la forma en que se calcula el numero de recetas y la fecha[7] por la cual se consulta:

```
(select IFNULL(sum( if(
convert(DATEDIFF (proxEnt, ent)/30, signed int)=0, 1, convert(DATEDIFF (proxEnt, ent)/30, signed int) )),0) as receta
from
    (select distinct receta_farmacia.folio as rec,
        (select max(detalle_tarjeta_farmacia.fecha_prox_entrega)
        from detalle_tarjeta_farmacia
        where detalle_tarjeta_farmacia.folio_receta_farmacia= rec) as proxEnt,
        receta_farmacia.fecha as ent
    from receta_farmacia inner join detalle_tarjeta_farmacia on
        receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
        inner join paciente on paciente.run = receta_farmacia.run_paciente
```

  →

```
(select IFNULL(sum(if( convert (DATEDIFF (detalle_tarjeta_farmacia.fecha_prox_entrega, detalle_tarjeta_farmacia.fecha_entrega)/30, signed int)=0, 1,
convert (DATEDIFF (detalle_tarjeta_farmacia.fecha_prox_entrega, detalle_tarjeta_farmacia.fecha_entrega)/30, signed int))),0)
from
    detalle_tarjeta_farmacia inner join receta_farmacia on
    receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    inner join paciente on paciente.run = receta_farmacia.run_paciente
```

[7] La fecha de la Tabla detalle_tarjeta_farmacia representa la fecha en que se entrega el producto y la fecha de la Tabla de receta_farmacia representa la fecha en que se emite la receta.

```
and cast(receta_farmacia.fecha as date) between $P{fecha1} and $P{fecha2}
```

→

```
and detalle_tarjeta_farmacia.fecha_entrega between $P{fecha1} and $P{fecha2}
```

Haciendo estos y otros cambios se pueden generar las 22 sub-consultas necesarias para generar el sector 1 con grupo etario de 0 a 9 años mostrado anterior mente en la Figura 13. Las consultas y sub-consultas necesarias para generar los otros sectores y/o grupos etarios son similares por lo que basta con comprender los problemas de esta sección para comprender los problemas de las otras secciones.

La sección del resumen se genera con consultas similares pero no se considera la condición del sector. Como podemos observar se ejecutan muchas consultas, repitiendo muchos cálculos (o predicados) lo que provoca que tarde tanto tiempo en ser generado. A continuación se presentan algunas de las sub-consultas para generar este reporte:

- En la Figura 14 (b) presenta una sub-consulta similar pero que representa las prescripciones de los pacientes de morbilidad retiradas en horario diurno y que no pertenecen a ningún sector ("sin sector" en el reporte). Nuevamente se presenta la clasificación para el grupo etario como una variable para luego iterar esta sub-consulta por cada grupo:
  ```
  and  receta_farmacia.id_clasificacion = $P{clas_id}
  ```
- En la Figura 14 (c) se muestra la sub-consulta perteneciente a la consulta que genera en la sección del resumen, las recetas y prescripciones pertenecientes al grupo etario 4 (65 años y mas) y representa las prescripciones de los pacientes crónicos en horario diurno. En esta consulta no se considera el sector.
- En la Figura 14 (d) se muestra la sub-consulta que obtiene las recetas de los pacientes de procedencia SAPU en horario diurno. Esta sub-consulta pertenece a la última sección del resumen durante toda la consulta de esta sección no se consideran grupos etarios ni sector.

```
(select IFNULL(sum(if( convert (DATEDIFF (detalle_tarjeta_farmacia.fecha_prox_entrega, detalle_tarjeta_farmacia.fecha_entrega)/30, signed int)=0, 1,
convert (DATEDIFF (detalle_tarjeta_farmacia.fecha_prox_entrega, detalle_tarjeta_farmacia.fecha_entrega)/30, signed int))),0)
from
    detalle_tarjeta_farmacia inner join receta_farmacia on
    receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    inner join paciente on paciente.run = receta_farmacia.run_paciente
    and  substr(paciente.numFamilia, 2, 1)<> '1'
    and substr(paciente.numFamilia, 2, 1)<> '2'
    and substr(paciente.numFamilia, 2, 1)<> '3'
    and substr(paciente.numFamilia, 2, 1)<> '4'
    and substr(paciente.numFamilia, 2, 1)<> '5'
    and substr(paciente.numFamilia, 2, 1)<> '6'
    and  receta_farmacia.id_clasificacion = $P{clas_id}
    and detalle_tarjeta_farmacia.procedencia='Morbilidad'
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and detalle_tarjeta_farmacia.entregado=1
    and detalle_tarjeta_farmacia.fecha_entrega between $P{fecha1} and $P{fecha2}
    and detalle_tarjeta_farmacia.hora<'17:00:00'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)<>'Friday'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)<>'Saturday'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)<>'Sunday'

    or receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    and paciente.run = receta_farmacia.run_paciente
    and  substr(paciente.numFamilia, 2, 1)<> '1'
    and substr(paciente.numFamilia, 2, 1)<> '2'
    and substr(paciente.numFamilia, 2, 1)<> '3'
    and substr(paciente.numFamilia, 2, 1)<> '4'
    and substr(paciente.numFamilia, 2, 1)<> '5'
    and substr(paciente.numFamilia, 2, 1)<> '6'
    and  receta_farmacia.id_clasificacion = $P{clas_id}
    and detalle_tarjeta_farmacia.procedencia='Morbilidad'
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and detalle_tarjeta_farmacia.entregado=1
    and detalle_tarjeta_farmacia.fecha_entrega between $P{fecha1} and $P{fecha2}
    and detalle_tarjeta_farmacia.hora<'16:00:00'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)='Friday'

    or
    receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    and paciente.run = receta_farmacia.run_paciente
    and  substr(paciente.numFamilia, 2, 1)<> '1'
    and substr(paciente.numFamilia, 2, 1)<> '2'
    and substr(paciente.numFamilia, 2, 1)<> '3'
    and substr(paciente.numFamilia, 2, 1)<> '4'
    and substr(paciente.numFamilia, 2, 1)<> '5'
    and substr(paciente.numFamilia, 2, 1)<> '6'
    and  receta_farmacia.id_clasificacion = $P{clas_id}
    and detalle_tarjeta_farmacia.procedencia='Morbilidad'
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and detalle_tarjeta_farmacia.entregado=1
    and detalle_tarjeta_farmacia.fecha_entrega between $P{fecha1} and $P{fecha2}
    and detalle_tarjeta_farmacia.hora<'9:00:00'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)='Saturday'

    or receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    and paciente.run = receta_farmacia.run_paciente
    and  substr(paciente.numFamilia, 2, 1)<> '1'
    and substr(paciente.numFamilia, 2, 1)<> '2'
    and substr(paciente.numFamilia, 2, 1)<> '3'
    and substr(paciente.numFamilia, 2, 1)<> '4'
    and substr(paciente.numFamilia, 2, 1)<> '5'
    and substr(paciente.numFamilia, 2, 1)<> '6'
    and  receta_farmacia.id_clasificacion = $P{clas_id}
    and detalle_tarjeta_farmacia.procedencia='Morbilidad'
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and detalle_tarjeta_farmacia.entregado=1
    and detalle_tarjeta_farmacia.fecha_entrega between $P{fecha1} and $P{fecha2}
    and detalle_tarjeta_farmacia.hora<'9:00:00'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)='Sunday') as diurno_Morbilidad,
```

Figura 14 (b): Prescripciones de Paciente Morbilidad en Horario Diurno y que no Pertenecen a Ningún Sector.

```
(select IFNULL(sum(if( convert (DATEDIFF (detalle_tarjeta_farmacia.fecha_prox_entrega, detalle_tarjeta_farmacia.fecha_entrega)/30, signed int)=0, 1,
convert (DATEDIFF (detalle_tarjeta_farmacia.fecha_prox_entrega, detalle_tarjeta_farmacia.fecha_entrega)/30, signed int))),0)
from
    detalle_tarjeta_farmacia inner join receta_farmacia on
    receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    inner join paciente on paciente.run = receta_farmacia.run_paciente
    and detalle_tarjeta_farmacia.procedencia='Crónico'
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and receta_farmacia.id_clasificacion= 4
    and detalle_tarjeta_farmacia.fecha_entrega between $P{fecha1} and $P{fecha2}
    and detalle_tarjeta_farmacia.hora>='17:00:00'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)<>'Friday'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)<>'Saturday'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)<>'Sunday'

    or receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    and paciente.run = receta_farmacia.run_paciente
    and detalle_tarjeta_farmacia.procedencia='Crónico'
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and receta_farmacia.id_clasificacion= 4
    and detalle_tarjeta_farmacia.fecha_entrega between $P{fecha1} and $P{fecha2}
    and detalle_tarjeta_farmacia.hora>='16:00:00'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)='Friday'

    or receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    and paciente.run = receta_farmacia.run_paciente
    and detalle_tarjeta_farmacia.procedencia='Crónico'
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and receta_farmacia.id_clasificacion= 4
    and detalle_tarjeta_farmacia.fecha_entrega between $P{fecha1} and $P{fecha2}
    and detalle_tarjeta_farmacia.hora>='9:00:00'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)='Saturday'

    or receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    and paciente.run = receta_farmacia.run_paciente
    and detalle_tarjeta_farmacia.procedencia='Crónico'
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and receta_farmacia.id_clasificacion= 4
    and detalle_tarjeta_farmacia.fecha_entrega between $P{fecha1} and $P{fecha2}
    and detalle_tarjeta_farmacia.hora>='9:00:00'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)='Sunday') as vespertino_cronico,
```

Figura 14 (c): Consulta de los Pacientes Crónicos en Horario Vespertino Pertenecientes al Grupo Etario 4.

```
(select IFNULL(sum( if(
convert(DATEDIFF (proxEnt, ent)/30, signed int)=0, 1, convert(DATEDIFF (proxEnt, ent)/30, signed int) )),0) as receta
from
    (select distinct receta_farmacia.folio as rec,
        (select max(detalle_tarjeta_farmacia.fecha_prox_entrega)
        from detalle_tarjeta_farmacia
        where detalle_tarjeta_farmacia.folio_receta_farmacia= rec) as proxEnt,
        receta_farmacia.fecha as ent
    from receta_farmacia inner join detalle_tarjeta_farmacia on
    receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    inner join paciente on paciente.run = receta_farmacia.run_paciente
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and detalle_tarjeta_farmacia.entregado=1
    and cast(receta_farmacia.fecha as date) between '2011-07-01' and '2011-07-31'
    and detalle_tarjeta_farmacia.procedencia= 'SAPU'
    and cast(receta_farmacia.fecha as time)<'18:00:00') as tab) as diurno_SAPU_receta,
```

Figura 14 (d): Resumen Receta Pacientes SAPU en Horario Diurno.

### 4.2.2 **Reporte Monitoreo Diario**

Este reporte al momento de ser solicitado tarda aproximadamente 15 minutos con 30 segundos, lo que para el Consultorio es un tiempo excesivo y desearía poder reducirlo a 5 minutos aprox.

El reporte “monitoreo diario” nos entrega un resumen de todas las recetas y prescripciones dentro de un mes determinado por el usuario, este se muestra en la Figura 15 donde se puede observar que existen cuatro secciones: (1) en la primera se muestran todas las recetas y prescripciones que se entregaron diariamente. (2) en la segunda sección se muestran todas las recetas y prescripciones que se entregaron y despacharon el mismo día. (3) en la tercera sección se muestra las recetas que no se despacharon (dispensaron) el mismo día que se entregaron. (4) en la última sección se muestra el porcentaje de recetas despachadas el mismo día. Como se puede observar en este reporte se muestran los pacientes de procedencia crónica y morbilidad, pero dentro de morbilidad son considerados los pacientes de procedencia SAPU.

Este reporte se relaciona con el reporte de estadística farmacia, por Ejemplo, si el rango de fechas escogidos para crear el reporte estadística farmacia corresponde a 01-05-2011 al 31-05-2011 y el mes escogido del reporte, monitoreo diario es el 05-2011, deben coincidir los totales de las recetas entregadas.

En este reporte se ejecuta 1 consulta por cada una de las 4 secciones, por día con 9; 6; 15 y 8 sub-consultas (cada sección), es decir, si consideramos un mes de 30 días se ejecutan 120 consultas con 1140 sub-consultas.

MES: 6
AÑO: 2011

MONITOREO DIARIO DE RECETAS EN ATENCIÓN PRIMARIA

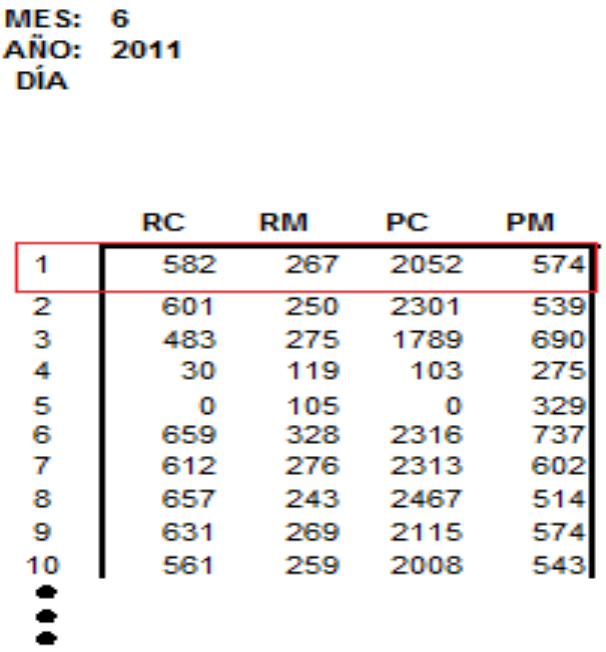
| DÍA | N° Rp despachadas antes de 24 horas | | | | N° Rp despachadas después de 24 horas | | | | % de recetas para pacientes ambulatorios dispensadas completas el mismo día de su emisión |
|---|---|---|---|---|---|---|---|---|---|
| | RC | RM | PC | PM | RC | RM | PC | PM | |
| 1 | 582 | 267 | 2052 | 574 | 132 | 226 | 433 | 504 | 450 | 41 | 1619 | 70 | 95,17 |
| 2 | 601 | 250 | 2301 | 539 | 148 | 219 | 527 | 495 | 453 | 31 | 1774 | 44 | 96,36 |
| 3 | 483 | 275 | 1789 | 690 | 144 | 221 | 554 | 587 | 339 | 54 | 1235 | 103 | 92,88 |
| 4 | 30 | 119 | 103 | 275 | 17 | 113 | 69 | 262 | 13 | 6 | 34 | 13 | 95,97 |
| 5 | 0 | 105 | 0 | 329 | 0 | 100 | 0 | 312 | 0 | 5 | 0 | 17 | 95,24 |
| 6 | 659 | 328 | 2316 | 737 | 156 | 274 | 474 | 658 | 503 | 54 | 1842 | 79 | 94,53 |
| 7 | 612 | 276 | 2313 | 602 | 134 | 236 | 482 | 542 | 478 | 40 | 1831 | 60 | 95,5 |
| 8 | 657 | 243 | 2467 | 514 | 225 | 198 | 877 | 430 | 432 | 45 | 1590 | 84 | 95 |
| 9 | 631 | 269 | 2115 | 574 | 175 | 227 | 625 | 513 | 456 | 42 | 1490 | 61 | 95,33 |
| 10 | 561 | 259 | 2008 | 543 | 171 | 200 | 699 | 434 | 390 | 59 | 1309 | 109 | 92,8 |
| 11 | 15 | 143 | 56 | 372 | 2 | 136 | 2 | 362 | 13 | 7 | 54 | 10 | 95,57 |
| 12 | 0 | 96 | 0 | 220 | 0 | 93 | 0 | 217 | 0 | 3 | 0 | 3 | 96,88 |
| 13 | 690 | 290 | 2525 | 647 | 185 | 250 | 698 | 562 | 505 | 40 | 1827 | 85 | 95,92 |
| 14 | 553 | 285 | 2122 | 670 | 185 | 250 | 799 | 612 | 368 | 35 | 1323 | 58 | 95,82 |
| 15 | 569 | 222 | 1946 | 429 | 169 | 177 | 523 | 369 | 400 | 45 | 1423 | 60 | 94,31 |
| 16 | 668 | 221 | 2482 | 459 | 144 | 176 | 509 | 385 | 524 | 45 | 1973 | 74 | 94,94 |
| 17 | 506 | 234 | 1609 | 510 | 83 | 185 | 224 | 434 | 423 | 49 | 1385 | 76 | 93,38 |
| 18 | 12 | 94 | 52 | 256 | 3 | 92 | 4 | 248 | 9 | 2 | 48 | 8 | 98,11 |
| 19 | 0 | 79 | 0 | 155 | 0 | 75 | 0 | 147 | 0 | 4 | 0 | 8 | 94,94 |
| 20 | 722 | 241 | 2617 | 453 | 158 | 205 | 578 | 394 | 564 | 36 | 2039 | 59 | 96,26 |
| 21 | 653 | 281 | 2536 | 613 | 171 | 234 | 752 | 531 | 482 | 47 | 1784 | 82 | 94,97 |
| 22 | 609 | 274 | 2191 | 608 | 200 | 234 | 684 | 538 | 409 | 40 | 1507 | 70 | 95,47 |
| 23 | 695 | 196 | 2562 | 418 | 158 | 158 | 521 | 362 | 537 | 38 | 2041 | 56 | 95,74 |
| 24 | 590 | 218 | 2117 | 451 | 166 | 179 | 636 | 390 | 424 | 39 | 1481 | 61 | 95,17 |
| 25 | 12 | 127 | 31 | 280 | 0 | 107 | 0 | 237 | 12 | 20 | 31 | 43 | 85,61 |
| 26 | 0 | 104 | 0 | 279 | 0 | 94 | 0 | 261 | 0 | 10 | 0 | 18 | 90,38 |
| 27 | 2 | 94 | 2 | 269 | 1 | 89 | 1 | 259 | 1 | 5 | 1 | 10 | 94,79 |
| 28 | 598 | 285 | 2185 | 624 | 149 | 230 | 646 | 543 | 449 | 55 | 1539 | 81 | 93,77 |
| 29 | 577 | 269 | 1966 | 580 | 196 | 225 | 667 | 497 | 381 | 44 | 1299 | 83 | 94,8 |
| 30 | 476 | 212 | 1516 | 473 | 98 | 175 | 189 | 414 | 378 | 37 | 1327 | 59 | 94,62 |

Figura 15: Reporte Monitoreo Diario.

La consulta para generar la primera sección de este reporte (Figura 16) que muestra las recetas de pacientes crónicos (RC), las recetas de los pacientes de morbilidad (RM), las prescripciones de los pacientes crónicos (PC) y las prescripciones de los pacientes de morbilidad (PM), se estructura de la misma forma como se mencionó anteriormente en el reporte de estadística Farmacia (Sección 4.2.1) pero no se consideran sectores ni grupos etarios. Para obtener el total diario de las recetas prescripciones se calculan las recetas vespertinas y diurnas de los pacientes de procedencia crónica, morbilidad y SAPU, y luego se suman. Y para las prescripciones se calculan sin distinguir entre vespertino y diurno, para crónicos,

SAPU y morbilidad. Luego para obtener los totales de morbilidad se calcula el total de morbilidad y SAPU (recetas y prescripciones), y luego se suman. Esta consulta se itera por cada día del mes consultado. En este caso no se consideran las clausulas para el sector y el grupo etario. Un ejemplo de una sub-consulta que calcula el número de recetas de pacientes de morbilidad diurno, se muestra en la Figura 17. Luego de obtener este resultado se debe calcular el de recetas morbilidad vespertina, el de recetas SAPU diurna y vespertina, para efectuar la suma ya mencionada.

Por lo que esta consulta presenta los mismos problemas presentados en el reporte de estadística farmacia.



MES: 6
AÑO: 2011
DÍA

| | RC | RM | PC | PM |
|---|---|---|---|---|
| 1 | 582 | 267 | 2052 | 574 |
| 2 | 601 | 250 | 2301 | 539 |
| 3 | 483 | 275 | 1789 | 690 |
| 4 | 30 | 119 | 103 | 275 |
| 5 | 0 | 105 | 0 | 329 |
| 6 | 659 | 328 | 2316 | 737 |
| 7 | 612 | 276 | 2313 | 602 |
| 8 | 657 | 243 | 2467 | 514 |
| 9 | 631 | 269 | 2115 | 574 |
| 10 | 561 | 259 | 2008 | 543 |

Figura 16: Primera Sección Monitoreo Diario.

```
(select IFNULL(sum( if(
convert(DATEDIFF (proxEnt, ent)/30, signed int)=0, 1, convert(DATEDIFF (proxEnt, ent)/30, signed int) )),0) as receta
from
    (select distinct receta_farmacia.folio as rec,
        (select max(detalle_tarjeta_farmacia.fecha_prox_entrega)
        from detalle_tarjeta_farmacia
        where detalle_tarjeta_farmacia.folio_receta_farmacia= rec) as proxEnt,
    receta_farmacia.fecha as ent
    from
    receta_farmacia inner join detalle_tarjeta_farmacia on
    receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and detalle_tarjeta_farmacia.entregado=1
    and cast(receta_farmacia.fecha as date)=$P{fecha}
    and detalle_tarjeta_farmacia.procedencia= 'Morbilidad'
    and cast(receta_farmacia.fecha as time)<'17:00:00'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)<>'Friday'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)<>'Saturday'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)<>'Sunday'

    or receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    and detalle_tarjeta_farmacia.procedencia='Morbilidad'
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and cast(receta_farmacia.fecha as date)=$P{fecha}
    and detalle_tarjeta_farmacia.entregado=1
    and cast(receta_farmacia.fecha as time)<'16:00:00'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)='Friday'

    or receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    and detalle_tarjeta_farmacia.entregado=1
    and detalle_tarjeta_farmacia.procedencia='Morbilidad'
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and cast(receta_farmacia.fecha as date)=$P{fecha}
    and cast(receta_farmacia.fecha as time)<'9:00:00'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)='Saturday'

    or receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    and detalle_tarjeta_farmacia.entregado=1
    and detalle_tarjeta_farmacia.procedencia='Morbilidad'
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and cast(receta_farmacia.fecha as date)=$P{fecha}
    and cast(receta_farmacia.fecha as time)<'9:00:00'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)='Sunday' ) as tab) as receta_morbi_diurno,
```

Figura 17: Sub-consulta Receta Morbilidad Diurna.

Para la siguiente sección de este reporte “N° Rp despachadas antes de 24 horas” se utiliza una consulta parecida a las vistas anteriormente, combinada con la relación de reserva hora. Si existe una hora reservada para el mismo día, relacionada con el paciente de la receta, entonces se despacho dentro de las 24 horas. La Figura 18 muestra una sub-consulta que genera las recetas de procedencia de morbilidad.

```
(select IFNULL(sum( if(
convert(DATEDIFF (proxEnt, ent)/30, signed int)=0, 1, convert(DATEDIFF (proxEnt, ent)/30, signed int) )),0) as receta_dentro_24_horas
from
    (select distinct receta_farmacia.folio as rec,
        (select max(detalle_tarjeta_farmacia.fecha_prox_entrega)
        from detalle_tarjeta_farmacia
        where detalle_tarjeta_farmacia.folio_receta_farmacia= rec) as proxEnt,
    receta_farmacia.fecha as ent
    from
    receta_farmacia inner join detalle_tarjeta_farmacia on
    receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and detalle_tarjeta_farmacia.entregado=1
    and cast(receta_farmacia.fecha as date)=$P{fecha}
    and detalle_tarjeta_farmacia.procedencia= 'Morbilidad'
    and EXISTS
        (select reservahora.id_profesional
        from reservahora
        where reservahora.fecha= detalle_tarjeta_farmacia.fecha_entrega
        and reservahora.id_profesional= receta_farmacia.run_profesional
        and reservahora.id_paciente= receta_farmacia.run_paciente)
    or receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and detalle_tarjeta_farmacia.entregado=1
    and cast(receta_farmacia.fecha as date)=$P{fecha}
    and detalle_tarjeta_farmacia.procedencia= 'Morbilidad'
    and EXISTS
        (select control_sapu.folio
        from control_sapu
        where CAST(control_sapu.fecha_ingreso AS date) = detalle_tarjeta_farmacia.fecha_entrega
        and control_sapu.id_paciente = receta_farmacia.run_paciente)) as tab) as receta_morbilidad,
```

Figura 18: Recetas Morbilidad Despachadas Dentro de 24 horas.

Al igual que en el reporte de estadística farmacia existen muy pocas diferencias entre las consultas y sub-consultas necesarias para calcular esta sección. A continuación se presentan las diferencias entre esta sub-consulta y las demás sub-consultas de esta sección:

- Para obtener los resultados de las distintas procedencias se ejecuta esta sub-consulta modificando:
  `and detalle_tarjeta_farmacia.procedencia= 'Morbilidad'`
  → `and detalle_tarjeta_farmacia.procedencia='SAPU'` O por procedencia crónica.

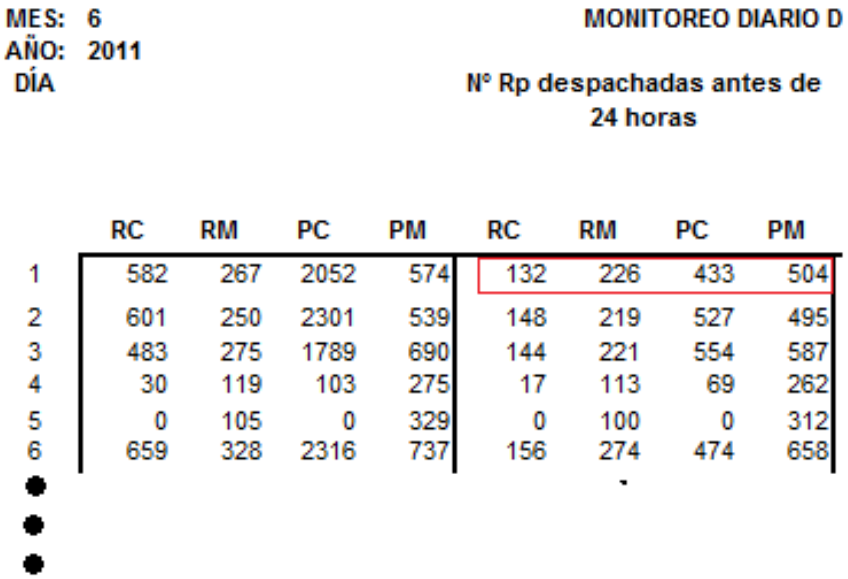
- Para obtener las prescripciones intercambiar de esta sub-consulta:

```
(select IFNULL(sum( if(
convert(DATEDIFF (proxEnt, ent)/30, signed int)=0, 1, convert(DATEDIFF (proxEnt, ent)/30, signed int) )),0) as receta_dentro_24_horas
from
    (select distinct receta_farmacia.folio as rec,
        (select max(detalle_tarjeta_farmacia.fecha_prox_entrega)
        from detalle_tarjeta_farmacia
        where detalle_tarjeta_farmacia.folio_receta_farmacia= rec) as proxEnt,
    receta_farmacia.fecha as ent
    from
    receta_farmacia inner join detalle_tarjeta_farmacia on
    receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
```

→

```
(
select IFNULL(sum(if( convert (DATEDIFF (detalle_tarjeta_farmacia.fecha_prox_entrega, detalle_tarjeta_farmacia.fecha_entrega)/30, signed int)=0, 1,
convert (DATEDIFF (detalle_tarjeta_farmacia.fecha_prox_entrega, detalle_tarjeta_farmacia.fecha_entrega)/30, signed int))),0) as prescripcion_cronico
from
    detalle_tarjeta_farmacia inner join receta_farmacia on
    receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
```

Cambiando y combinando estas instrucciones se obtienen las 6 sub consultas para generar la parte del reporte marcado por un cuadro rojo en la Figura 19. Luego esta consulta con sus sub-consultas son iteradas por cada día del mes escogido. Es importante recordar que para obtener los valores totales de morbilidad se suman las de procedencia de morbilidad con SAPU motivo por el cual son 6 sub-consultas y solo se presentan 4 resultados.



MES: 6
AÑO: 2011
DÍA

MONITOREO DIARIO D

N° Rp despachadas antes de 24 horas

| | RC | RM | PC | PM | RC | RM | PC | PM |
|---|---|---|---|---|---|---|---|---|
| 1 | 582 | 267 | 2052 | 574 | 132 | 226 | 433 | 504 |
| 2 | 601 | 250 | 2301 | 539 | 148 | 219 | 527 | 495 |
| 3 | 483 | 275 | 1789 | 690 | 144 | 221 | 554 | 587 |
| 4 | 30 | 119 | 103 | 275 | 17 | 113 | 69 | 262 |
| 5 | 0 | 105 | 0 | 329 | 0 | 100 | 0 | 312 |
| 6 | 659 | 328 | 2316 | 737 | 156 | 274 | 474 | 658 |

Figura 19: Recetas y Prescripciones Despachadas Antes de 24 Horas.

Para la tercera sección "N° Rp despachadas después de 24 horas" se ejecutan 15 sub-consultas que corresponden a las 9 sub-consultas de la primera sección con las 6 sub-consultas de la segunda sección, luego se realiza la resta correspondiente para obtener el valor requerido (por Ejemplo: receta crónicos despachados fuera de 24 horas = receta crónicos – crónicos despachados dentro de las 24 horas) y luego se iteran por cada día del mes escogido, finalmente para la última sección se realizan 8 sub-consultas donde se calculan todas las recetas (crónico, morbilidad y SAPU) y las recetas despachadas fuera de las 24 horas de los pacientes crónicos, divididas el total de las recetas de la primera sección (total recetas crónicos, morbilidad y SAPU) y multiplicadas por 100 para sacar el porcentaje correspondiente. Luego esta consulta se itera por cada día correspondiente al mese escogido.

En la Tabla número 3 se muestran valores estadísticos de cada reporte, como por ejemplo, el número de consultas y sub-consultas ejecutadas, tiempo promedio que tarda en ser obtenido y líneas de códigos promedio por consulta

| | **Estadística farmacia** | **Monitoreo diario** |
|---|---|---|
| Tiempo promedio de obtención | 8:30 min. | 15:30 min. |
| Número de Tablas involucradas | 4 | 4 |
| Consultas ejecutadas | 37 (veces) | 120 (veces) |
| Sub-consultas ejecutadas | 888 (veces) | 1140 (veces) |
| Líneas de código por consulta (promedio) | 1112 | 278 |

Tabla 3: Resumen Estadístico de los Reportes.

### *4.2.3* Otros Problemas Detectados

El rendimiento de la base de datos se puede ver afectado por otros motivos. Los siguientes son problemas que afectan los tiempos de respuestas al momento de ejecutar consultas.

**Un mal uso de los índices**, en la tabla paciente y en la tabla receta_farmacia sus claves primarias son de tipo varchar() (paciente.run = varchar(13) y receta_farmacia.folio = varchar(20)), según el manual de MySQL 5 es recomendable (cuando se puede) tener claves primarias de tipo numérico, esto facilita su búsqueda y comparación por ejemplo al hacer un join. Además esto ha provocado problemas de inconsistencia dentro de la base de datos, por ejemplo en la tabla paciente encontramos el run = '00886031-9' y el run = '0886031-9' lo cual crea duplicidad de datos para un mismo paciente.

**Las recetas** (físicas) con las que se trabaja tienen su propio folio y cada doctor tiene su taco (o talonario) de recetas por lo que existe la posibilidad que dos recetas tengan el mismo folio, en estos casos el sistema debe actualizar la tabla

receta_farmacia modificando el folio de la receta existente, por ejemplo si se desea agregar una receta con el folio "000164" pero este folio ya existe entonces el sistema actualiza la receta ingresada anteriormente dejando su folio como "000164-1" y luego insertando la receta deseada.

## 4.3 Propuesta y Objetivos del Proyecto

Debido al previo análisis e hipótesis inicial se espera mejorar los tiempos y rendimiento del sistema de control de farmacia realizando un estudio sobre:

- El procesamiento de consultas de base de datos.
- La Optimización de base de datos basa en heurissticas.
- Optimización de múltiples consultas.

Luego de analizar los casos estudiados éstos serán aplicados y adaptados (si es necesario) para dar solución a los problemas planteados en la Sección 4.2 y para cumplir con los objetivo propuestos en la Sección 1.1, enfocados siempre en el motor de base de datos MySQL 5.0 "*Puede que pequeños cambios pueden significar la ganancia de muchos segundos o minutos que el usuario debe esperar al momento de ejecutar la consulta*." (Oracle and/or its affiliates, 2011). Si bien lo primordial de esta investigación son las estrategias de ejecución es necesario que las consultas con que el sistema trabaja (donde se espera ejecutar la estrategia) sean correctas y óptimas.

# Capítulo V: Procesamiento y Optimización de Consultas

## 5.1 Introducción

Cuando se presenta una consulta de alto nivel al sistema el SGMD, necesita encontrar el mejor método para entregar la respuesta utilizando la estructura de base de datos existente, los pasos necesarios para encontrar una estrategia adecuada durante el procesamiento de la consulta se muestran en la Figura 20. Esta consulta expresada en un lenguaje de alto nivel como SQL, primero debe pasar por un **análisis léxico,** un **análisis sintáctico** y una **validación**. El analizador léxico identifica los símbolos o palabras claves del lenguaje (como las palabras claves de SQL, los nombres de los atributos y los nombres de las relaciones) en el contexto de la consulta, mientras que el analizador sintáctico revisa la sintaxis de la consulta para determinar si está formulada de acuerdo a las reglas sintácticas del lenguaje de consulta. Además la consulta se debe validar para lo cual ha de comprobarse que todos los nombres de atributos y relaciones sean válidos y tengan sentido desde el punto sistemático en el esquema de base de datos concreta que se esté consultando. A continuación, se crea una representación interna (o intermedia) de la consulta, por lo general en forma de estructura de árbol de datos o grafo de datos denominado "árbol de consulta" y "grafo de consulta" respectivamente. Seguidamente, el SGBD debe optimizar esta consulta. El optimizador recibe como entrada un árbol de consulta A y genera un árbol de consulta B (que representa un plan de ejecución) equivalente al árbol de consulta A pero más eficiente, ejecutando este plan de ejecución óptimo se obtiene el resultado de la consulta (Elmasri & Navathe, 2002).

Este capítulo se centra en el proceso de la optimización abordando los distintos tipos de optimización que son utilizados para dar solución a los problemas planteados en la sección 4.2

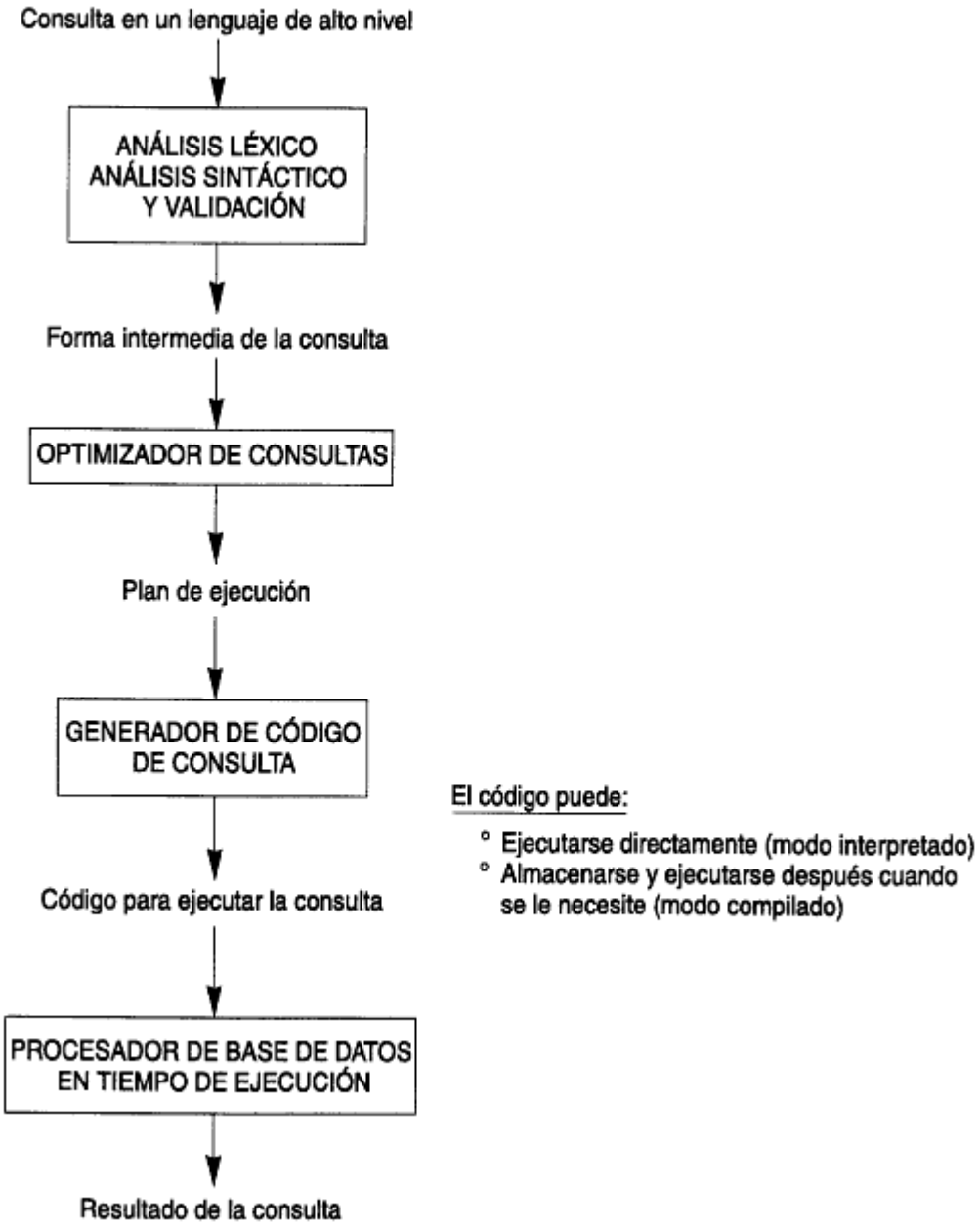



Figura 20: Pasos del Procesamiento de una Consulta de Alto Nivel. (Elmasri & Navathe, 2002)

### 5.1.1 Ejemplo Motivador

Existe un gran número de estrategias de ejecución posibles para una determinada consulta y la diferencia entre una buena y una mala estrategia se basa en el costo del procesamiento de cada consulta, normalmente este costo es determinado por el número de accesos al disco y el proceso de seleccionar la estrategia de ejecución más adecuada se denomina optimización de consultas. A su vez, el tiempo empleado en elegir una estrategia de procesamiento de consultas merece la pena incluso para una consulta que se ejecutará una solo una vez. En el siguiente ejemplo podemos ver la diferencia, a grandes magnitudes, entre dos estrategias diferentes para una misma consulta:

**Ejemplo:**

Recordemos el ejemplo "Obtención de códigos de los productos" en la Sección 3.3.7 que consistía en obtener el código de los productos prescritos del paciente con el Rut = '9.078.656-3'. La consulta en un lenguaje de alto nivel correspondía a:

```
SELECT codigo_producto
FROM receta_farmacia, detalle_tarjeta_farmacia
WHERE receta_farmacia.folio=detalle_tarjeta_farmacia.folio_receta_farmacia
AND receta_farmacia.run_paciente = '9078656-3'
```

Para esta consulta obtuvimos una expresión equivalente en el álgebra relacional, la cual llamaremos estrategia 1, sin embargo existen varias estrategias posibles por ejemplo observemos la estrategia 2:

Estrategia 1:

$$\boldsymbol{\Pi}_{codigo\_producto}\,(\boldsymbol{\sigma}_{run\_paciente = 9078656\text{-}3 \,\wedge\, receta\_farmacia.folio = detalle\_tarjeta\_farmacia.folio\_receta\_farmacia}\,(receta\_farmacia \times detalle\_tarjeta\_farmacia))$$

Estrategia 2:

$$\boldsymbol{\Pi}_{codigo\_producto}\,(detalle\_tarjeta\_farmacia \bowtie (\boldsymbol{\sigma}_{run\_paciente = 9078656\text{-}3}\,(receta\_farmacia)))$$

Como podemos observar la estrategia 2 evita el producto cartesiano por lo se considera "mejor". Debido al gran número de factores que influyen en la evaluación de las consultas, la mayoría de las técnicas depende en gran medida en el análisis heurístico. Es importante también conocer cómo funciona un SGBD por lo que en el anexo E: algoritmos básicos se mencionan Algoritmos básicos para ejecutar operaciones de consulta como por ejemplo la implementación de la operación selección explicando el uso de índices para el acceso a los datos.

### 5.1.2 Optimización de Consultas en Asuntos Prácticos

La optimización de consultas según (Abiteboul, Hull, & Vianu, 1995) es uno de los temas centrales de los sistemas de base de datos. Una gran cantidad de factores juegan un papel importante en esta área, incluyendo las técnicas de almacenamiento e indexación, los tamaños de página y los protocolos de búsqueda, el sistema operativo subyacente, las propiedades estadísticas de las consultas previas y actualizaciones, y el poder expresivo de los lenguajes de consultas utilizados, por nombrar algunos entre otros.

La optimización de consultas se puede realizar en todos los niveles de la arquitectura de base de datos de tres niveles. Este trabajo se centra en el nivel físico, por ejemplo las técnicas de acceso a los datos y la gestión de buffer. A un nivel más lógico se utilizan equivalencias algebraicas para rescribir de manera más eficiente las consultas.

El supuesto habitual de bases de datos relacionales es que el estado actual de la base de datos es tan grande, que debe ser almacenado en la memoria secundaria (por ejemplo, en el disco). La manipulación de los datos almacenados, incluyendo la aplicación de operadores algebraicos, requiere la realización de copias en la memoria principal de las porciones de los datos almacenados y almacenar resultados intermedios y finales de nuevo en la memoria secundaria. Por lo que el gasto de tiempo en el procesamiento de consultas es bastante significativo sobre todo para un sistema singleprocessor. Como veremos, los optimizadores de consultas suelen desarrollar varios planes de evaluación y luego elegir uno para su ejecución.

## 5.2 Heurísticas en la Optimización de Consultas

Las reglas heurísticas son utilizadas para modificar la representación interna de una consulta, que suele estar en forma de estructura de datos árbol o grafo, a fin de mejorar el rendimiento esperado durante la ejecución.

### 5.2.1 Optimización Heurística de Árboles de Consulta

Como ya se ha mencionado, pueden existir varias expresiones del álgebra relacional para una misma consulta por lo que también existen varios árboles de consulta para una misma consulta. El analizador **sintáctico** de consultas casi siempre genera un árbol de consulta inicial estándar que corresponde a una consulta en SQL, sin efectuar optimización alguna, en este punto el optimizador de consultas heurístico debe transformar ese árbol de consulta inicial en un árbol de consulta final cuya ejecución sea eficiente.

El optimizador debe incluir reglas de equivalencia entre expresiones del álgebra relacional que puedan aplicarse al árbol inicial. Las reglas de heurísticas de

optimización de consultas utilizan estas expresiones de equivalencia para transformar el árbol inicial en el árbol de consulta final optimizado.

### 5.2.2 Reglas Generales de Transformación para Operaciones del Álgebra Relacional

Hay muchas reglas para transformar operaciones del álgebra relacional en otras equivalentes, en este caso lo que realmente interesa es el significado de las operaciones y las relaciones resultantes. Por lo tanto, si dos relaciones tienen el mismo conjunto de atributos en un orden distinto, pero ambas representan la misma información, se considera equivalente (cambio sintáctico). Algunas reglas de transformación útiles mencionadas por (Tamer Ozsu & Valduriez, 1999) y (Elmasri & Navathe, 2002) se destacan a continuación:

#### *5.2.2.1 Reglas de Transformación*

Ídem potencia de operadores unarios:

1. Una secuencia de restricciones sobre una tabla A puede transformarse en una sola restricción:
   $\sigma_{C1}(\sigma_{C2}(A)) \equiv \sigma_{C1 \text{ AND } C2}(A)$
2. En una secuencia de proyecciones contra una tabla A pueden ignorarse todas, salvo la última (si cada columna mencionada en la última, también aparece en las demás):
   $\Pi_{P2}(\Pi_{P1}(A)) \equiv \Pi_{P2}(A)$, donde $P2 \subseteq P1$

Ídem potencia de una selección con una proyección:

1. Una restricción de una proyección puede transformarse en una proyección de una restricción:
   $\sigma_C(\Pi_P(A)) \equiv \Pi_P(\sigma_C(A))$
   Esto pude ser una buena idea pues se reduce el número de filas antes de la proyección.

Distributividad en operaciones binarias. Sea f un operador unario y $\otimes$ un operador binario, entonces f es distributivo respecto de $\otimes$ si $f(A \otimes B) = f(A) \otimes f(B)$:

1. La selección es distributivo respecto de la UNIÓN, INTERSECCIÓN y DIFERENCIA:
   $\sigma_C(R \Theta S) \equiv (\sigma_C(R)) \Theta (\sigma_C(S))$, donde $\Theta \in \{\cup, \cap, -\}$
   Aquí se puede reducir el número de filas que serán examinadas en la siguiente operación.
2. La proyección es distributivo respecto de la UNIÓN:
   $\Pi_P(R \cup S) \equiv (\Pi_P(R)) \cup (\Pi_P(S))$

3. La selección es distributivo respecto de REUNIÓN (JOIN) si la condición de selección contiene columnas que sólo pertenecen a una tabla:
   $\sigma_{a=2}(R \bowtie_{br=bt} T) \equiv R \bowtie_{br=bt} (\sigma_{a=2}(T))$
   O puede escribirse como (c1 AND c2), y en c1 sólo intervienen columnas de R1 y en c2 sólo hay columnas de R2:
   $\sigma_c(R1 \bowtie_J R2) \equiv (\sigma_{c1}(R1)) \bowtie_J (\sigma_{c2}(R2))$
4. La proyección es distributivo respecto de JOIN si en la condición de reunión J sólo intervienen columnas incluidos en la lista de proyección P:
   $\Pi_P(R1 \bowtie_J R2) \equiv (\Pi_{P\_R1}(R1)) \bowtie_J (\Pi_{P\_R2}(R2))$, si y solo si P= (P_R1 $\cup$ P_R2) y P incluye toda columna de reunión que aparece en J.

Conmutatividad de selección con operaciones binarias:

1. $\sigma_{p(A)}(R \times S) \equiv (\sigma_{p(A)}(R)) \times S$
2. $\sigma_{p(Ai)}(R \bowtie_{(Aj,Bk)} S) \equiv (\sigma_{p(Ai)}(R)) \bowtie_{(Aj,Bk)} S$
3. $\sigma_{p(Ai)}(R \cup T) \equiv \sigma_{p(Ai)}(R) \cup \sigma_{p(Ai)}(T)$, Donde Ai pertenece a R y a T.

Conmutatividad de la proyección con operaciones binarias:

1. $\Pi_C(R \times S) \equiv \Pi_{A'}(R) \times \Pi_{B'}(S)$
2. $\Pi_C(R \bowtie_{(Aj,Bk)} S) \equiv \Pi_{A'}(R) \bowtie_{(Aj,Bk)} \Pi_{B'}(S)$
3. $\Pi_C(R \cup S) \equiv \Pi_C(R) \cup \Pi_C(S)$, Donde R[A] y S[B]; C = A' $\cup$ B' donde A' $\subseteq$ A, B' $\subseteq$ B

Asociatividad de las operaciones binarias:

1. $(R \times S) \times T \equiv R \times (S \times T)$
2. $(R \bowtie S) \bowtie T \equiv R \bowtie (S \bowtie T)$

Conversión de una secuencia ($\sigma$, $\times$) a $\bowtie$: si la condición c de operación $\sigma$ seguida de una $\times$ corresponde a una condición de reunión, convertir la secuencia ($\sigma$, $\times$) en $\bowtie$ de la manera siguiente:

1. $\sigma_c(R \times S) \equiv \sigma_c(R \bowtie S)$

Otras de las posibles reglas de transformación son las conocidas leyes DeMorgan:

1. NOT (c1 AND c2) $\equiv$ (NOT c1) OR (NOT c2)
2. NOT (c1 OR c2) $\equiv$ (NOT c1) AND (NOT c2)

Además algunas de estas transformaciones son recomendadas realizar según el manual de MySQL (Oracle and/or its affiliates, 2011), pero en esté se presentan estás transformaciones en un lenguaje de alto nivel y no en el álgebra relacional.

### 5.2.3 Características Principales

Al momento de realizar las transformaciones algebraicas se debe tener en cuenta las siguientes características de la optimización heurísticas:

1. Ejecutar operaciones de restricción σ tan pronto como sea posible.
2. Ejecutar primero las restricciones σ más restrictivas (las que producen menor nº de filas).
3. Ejecutar las operaciones de proyección Π tan pronto como sea posible.
4. Reordenar las operaciones para reducir el tamaño intermedio de relación.
5. Agrupar sub-expresiones comunes.
6. Remplazar una unión por una serie de semijoins.
7. Optimizar las operaciones individuales

### 5.2.4 Ejemplo 3: optimización heurística

Utilizando algunas de las reglas de transformación algebraicas optimizaremos una consulta de alto nivel aplicando las heurísticas. Consideremos las relaciones utilizadas en el Ejemplo 1:

- (1)**Receta_farmacia(**Folio,Fecha,Run_profesional,Run_paciente,Pendiente,…,)
- (2)**Detalle_tarjeta_farmacia(**Folio receta farmacia,Fecha entrega,Código pto ducto, Procedencia, ,…,)

Entonces se solicita: Encontrar el número de recetas entregadas por el profesional “08288691-5” y que sean de procedencia “Morbilidad” esto en un lenguaje de alto nivel podría representarse por:

```
SELECT COUNT(receta_farmacia.folio)
FROM receta_farmacia INNER JOIN detalle_tarjeta_farmacia ON
receta_farmacia.folio = detalle_tarjeta_farmacia.folio_receta_farmacia
WHERE detalle_tarjeta_farmacia.run_profecional=”08288691-5”
AND detalle_tarjeta_farmacia.procedencia=”Morbilidad”
```

Luego del análisis léxico, análisis sintáctico y la validación se obtiene una expresión interna, utilizando el álgebra relacional, obtenemos el árbol de consulta inicial (ver Figura 21) que luego será optimizado. La expresión en el álgebra relacional seria:

$\Pi_{Count(folio)}(\sigma_{run\_profecional="08288691-5"}(\sigma_{procedencia="Morbilidad"}(receta\_farmacia \bowtie_{folio=folio\_receta\_farmacia} detalle\_tarjeta\_farmacia))$

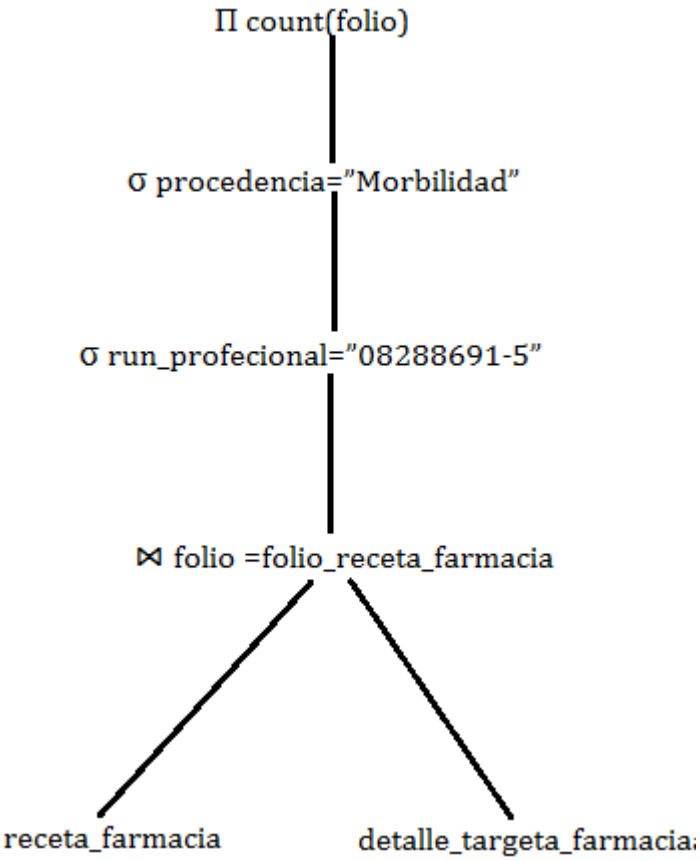



Figura 21: Árbol Inicial de Consulta del Ejemplo 2

A continuación se aplican y detallan las reglas de transformación, teniendo siempre en cuenta las características principales (Sección 5.2.3) de la optimización con heurísticas:

- Utilizando la conmutatividad de selección con operaciones binarias y considerando "Ejecutar operaciones de restricción σ tan pronto como sea posible" podemos reordenar la expresión de la siguiente manera: $\Pi_{Count(folio)}(\sigma_{run\_profecional="08288691-5"}(receta\_farmacia \bowtie_{folio=folio\_receta\_farmacia} \sigma_{procedencia="Morbilidad"} detalle\_tarjeta\_farmacia))$.
- Utilizando la misma regla anterior nuevamente tenemos: $\Pi_{Count(folio)}(\sigma_{run\_profecional="08288691-5"} receta\_farmacia \bowtie_{folio=folio\_receta\_farmacia} \sigma_{procedencia="Morbilidad"} detalle\_tarjeta\_farmacia)$.
- Ya se ha logrado una importante optimización pues se han reducido las relaciones intermedias ya que para realizar el JOIN solo se utilizaran las tuplas que se necesitas. Pero también podríamos reducir la cantidad de

columnas de cada tupla utilizando solo los atributos que necesitamos para llegar a nuestro resultado. Utilizaremos la conmutatividad de la proyección con operaciones binarias:

$\Pi_{Count(folio)}$ ($\Pi_{folio}$ ($\sigma_{run\_profecional="08288691-5"}$ receta_farmacia) $\bowtie_{folio=folio\_receta\_farmacia}$ $\Pi_{folio\_receta\_farmacia}$ ($\sigma_{procedencia="Morbilidad"}$ detalle_tarjeta_farmacia)).

Se necesita el folio de receta_farmacia pues es utilizado para el JOIN y el COUNT que generan el resultado y se necesita de detalle_tarjeta_farmacia el folio_receta_farmacia para el JOIN.

Llegando así al árbol de consulta final como muestra la Figura 22. Para la realización de la optimización algunos pasos se pueden obviar gracias a la utilización del árbol de consulta y trabajando sobre él.

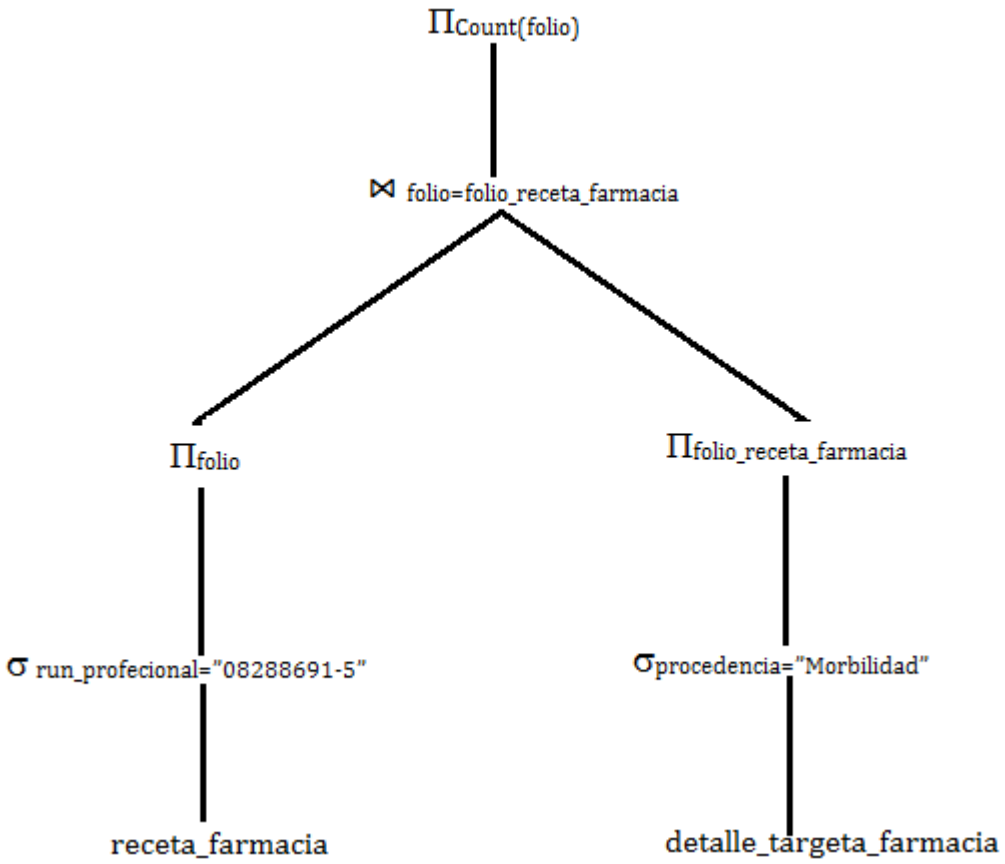



Figura 22: Árbol de Consulta Final.

## 5.3 Optimización de Múltiples Consultas

Dentro de los posibles escenarios de la optimización de los sistemas de bases de datos relacionales nos encontramos con uno en donde las consultas de varios usuarios se procesan en serie y algunas de esas consultas comparten relaciones de la base de datos. También existe el escenario en donde se procesa una consulta de tipo recursivo dividiéndola en múltiples consultas más pequeñas para simplificar el procesamiento. Además es importante recordar que SQL trabaja las consultas por bloques por lo que una sub-consulta se considera una consulta aparte.

Dentro de estos escenarios se presenta el problema de la optimización de consultas múltiples (o MQP de *multiple-query processing*).

Consideremos que se tiene una base de datos que consta de un conjunto de relaciones {R1, R2, . . . ,Rm}. Se tiene también un conjunto de consultas Q = {Q1, Q2, . . . , Qn} sobre varias de las relaciones de la base de datos.

El objetivo es buscar dentro de este conjunto de consultas un plan óptimo de acceso global, que permita recuperar la información necesaria para satisfacer cada una de las consultas en Q con el menor costo computacional posible (costo de CPU, costo de entrada y salida de datos, etc.). Este problema se podría abordar optimizando cada plan de ejecución para cada consulta del conjunto, utilizando técnicas de optimización como la optimización basa de heurísticas o semántica, esto se denomina *plan óptimo local;* si bien esto mejoraría el conjunto de consultas no garantiza que sea óptimo entre todos los planes globales, y al considerar el mínimo costo entre todos los planes globales se habla de *plan optimo global* una forma de poder realizar esto sería generar todas las posibles combinaciones de planes entre todas las consultas de forma que al hacer una suma de costos de todos los planes, esta suma sea menor o igual a una constante K correspondiente al costo mínimo parcial de los planes de acceso globales. Adicionalmente, los cálculos realizados en el primer caso los planes no tienen en cuenta sub-expresiones comunes entre las consultas, por lo que se pueden presentar cálculos repetidos.

El problema de la optimización de consultas múltiples (o plan optimo global) se puede dividir en dos partes: la primera parte es la optimización sobre la identificación de sub-expresiones y operaciones comunes entre las consultas, para así comprobar posibles beneficios de compartir los datos; la segunda parte hace referencia a optimizar la búsqueda de un plan global de acceso para un conjunto de consultas, de forma que el costo total de procesamiento de los sub-planes sea mínimo. Por Ejemplo, consideremos las siguientes dos consultas:

- C1: seleccionar todas las recetas despachadas el día 01-07-2011.

- C2: seleccionar todas las recetas de procedencia crónica despachadas el día 01-07-2011.

En la primera parte, que corresponde a la identificación de sub-expresiones y operaciones comunes, debemos identificar la operación de seleccionar las recetas que pertenecen a la fecha indicada. La segunda parte, para identificar el plan global, correspondería establecer que consulta ejecutar primero, en este caso se debería ejecutar primero la consulta C1 y luego al obtener ese resultado ejecutar C2.

Al solucionar el problema de sub-expresiones comunes entre las consultas se puede obtener grandes ganancias en la cantidad de cálculos realizados. Sin embargo dependiendo de la cantidad de datos a procesar, puede llegar el momento en que se cope el espacio de memoria reservado para los resultados intermedios de estos cálculos.

En esta sección definiremos múltiples grafos (multi-grafos) que representan múltiples consultas de SPJ (selección – proyección – join), estos grafos fueron definidos por (Chen & Dunham, 1998) . La transformación de estos múltiples grafos facilita la ejecución de las operaciones de SPJ y proporciona heurísticas para seleccionar las operaciones comunes para la ejecución del plan global.

### 5.3.1 Definición de grafos múltiples y su uso

Definición 1: un multi-grafo G (R, SE, JE) es un grafo con R nodos, y arcos SE y JE donde:

1. Un nodo, r ∈ R, del multi-grafo representa una relación o un resultado intermedio derivado de las operaciones relacionales.
2. Un arco de selección, se ∈ SE, representa una selección sobre una relación, este arco se denotan con el ID (identificador de la consulta) y la condición de selección.
3. Un arco de join, je ∈ JE, entre dos relaciones representa una operación de join, este arco se denotan con el ID (identificador) de la consulta y la condición de join.

Este multi-grafo es similar al expuesto en la Sección 3.5 pero este involucras más de una consulta, es por este motivo se utilizan los identificadores de cada consulta en cada arco, ya sea de selección o reunión.

El multi-grafo se modifica dinámicamente mientras la consulta se va ejecutando. Cuando una operación de relación (selección o join) se lleva acabo el nodo y el arco involucrado se transforman en un nuevo nodo. Este nodo representa la operación recién realizada sobre el nodo original. Al final de la secuencia de las

operaciones relacionales, sólo uno nodo representa el resultado final de la consulta original.

Existen cuatro posibilidades dentro de los puntos en común entre dos determinadas condiciones (por ejemplo la condición de selección). (1) Primera posibilidad "no tienen nada en común"; (2) "las condiciones son idénticas"; (3) "subsunción de una consulta a otra" (*subsumption*); (4) "superposición de una consulta sobre otra" (*overlap*). Ejemplo "x > 5" subsume a "x > 10" porque el resultado de "x > 5" puede ser utilizado para evaluar "x > 10". Un ejemplo para el caso de la superposición, tenemos que el predicado/condición "x > 10" se superpone sobre la condición "x < 20". En esta sección nos interesan las operaciones de selección, proyección y reunión (join) sobre las posibilidades (2), (3) y (4). Las condiciones para que se cumplan estas posibilidades (puntos en común), hacen referencia a seleccionar los predicados sobre los atributos del mismo nombre y sobre las mismas relaciones. Del mismo modo, con las condiciones de un join se refieren a los predicados de esté.

Al momento de modificar el multi-grafo, utilizando las reglas de transformación para las múltiples consultas (porque se ejecuta una operación), necesitamos ver cuales operaciones del conjunto de consultas se verán afectadas por la actual operación, tomando en cuenta las cuatro posibilidades. La operación que se ejecuta sobre el multi-grafo que se representa por arco o arista (i,j), conecta el nodo i con el nodo j, pudiendo afectar el nodo i y/o el nodo j, por lo que se debería aplicar la transformación correspondiente. Para aquellas consultas que no pueden ser beneficiadas por la actual operación, es decir, no utiliza el resultado obtenido por la reciente operación, el nodo i y/o el nodo j son mantenidos para las siguientes operaciones. En cuanto a las relaciones correspondientes, solo tenemos que realizar las operaciones idénticas una sola vez para todas las operaciones equivalentes. Para el caso de subsunción, la operación/condición más débil se ejecutara primero; por lo tanto, el resto de las operaciones de las subsunción pueden utilizar el resultado de la operación más débil. Para el caso de la superposición, necesitamos crear un *super-conjunto de operaciones*[8] que generara el resultado para todos los casos de superposición.

Dada una lista de consultas QL con operaciones de selección idénticas, la ejecución de las operaciones de selección idénticas deben reflejar los cambios para todas las consultas de QL. Por lo tanto después de la ejecución de la operación selección en R, un nuevo nodo n se generara para reflejar la ejecución de las operaciones de seleccionar idénticas. En QL, todas las operaciones de selección sobre

[8] Un super-conjunto es la operación cuyo resultado es la unión de los resultados de las operaciones relacionadas. Por Ejemplo, el super-conjunto para (R.A > 1 AND R.A < 3) y (R.A > 2 AND R.A < 4) es (R.A > 1 AND R.A < 4)

R cambiaran ahora sobre el nodo n, porque ahora el nodo R no existe para todas las consultas de QL. Las consultas en QL con join´s sobre el nodo R ahora harán referencia con su arco sobre el nodo n.

Para todas las consultas de QL con subsunción sobre las operaciones de selección sobre R, se ejecutara en primer lugar la operación/condición más débil (se pueden utilizar super-conjuntos). Se asume que el nodo n es generado a partir del super-conjunto de las operaciones de selección. Las consultas en QL con join´s sobre el nodo R ahora harán referencia con su arco sobre el nodo n.

Para el caso selección de superposición, la operación es similar a la de subsunción pero el super-conjunto de operaciones no es cualquiera de las operaciones que se superponen. El super-conjunto de operaciones es la unión de todas las consultas de selección que se superponen. El resto de la operación es similar al caso de las consultas de subsunción.

Esta técnica aplicada en las operaciones de join´s sobre el multi-grafo es similar a la operación de selección. La única diferencia es que los arcos de los join´s conectan dos diferentes nodos; por lo tanto, dos nodos involucrados con un arco de un join se convierten en un solo nodo n. Si ahí operaciones de selección sobre los dos nodos originales después de la operación del join, el arco de selección sobre los nodos del join cambiaran arcos de selección sobre el nuevo nodo n en lugar de uno de los nodos del join. También, en el caso de subsunción en los join´s la condición más fuerte del join cambiara de unirse de conectar los dos nodos originales a conectar con el nodo n.

Existen dos métodos para manejar las operaciones de proyección: (a) realizar las operaciones de proyección después de cada operación de selección y reunión (join). (b) dejar todas las operaciones de proyección para las etapas finales del plan ejecución de la consulta. En este caso se estudiara el caso (b).

Una de las funciones cruciales del procesamiento de múltiples consultas es detectar sub-expresiones comunes de una forma eficiente. Para cuando más de un conjunto de operaciones comunes existe se puede romper el empate para seleccionar una operación utilizando un conjunto de heurísticas de prioridad.

### 5.3.2 Heurísticas de MQP

(Chen & Dunham, 1998) Presentan una lista de heurísticas que indican en que orden se deben ejecutar los arcos o acciones pertenecientes a un conjunto de operaciones comunes en un multi-grafo de la MQP. A continuación se describen dichas heurísticas, ordenadas según su prioridad:

1. Seleccione un grupo de arcos de selección, pertenecientes al mutigrafo, en donde su punto en común sea del tipo idéntico y que actúan sobre el mismo nodo.
2. Seleccione un grupo de arcos de selección, tal que sus condiciones de selección sean del tipo subsunción y liego ejecute una después de otra (como se explicó en la sección anterior).
3. Seleccione un grupo de arcos de selección tal que sus condiciones de selección sean del tipo de superposición crea un super-conjunto y ejecutar los arcos sobre esté (como se explico en la sección anterior).
4. Seleccione un grupo de arcos de condición de reunión (join) tal que sus condiciones del join sean idénticas.
5. Seleccione un grupo de arcos de join tal que sus condiciones del join sean del tipo subsunción.
6. Dejar para el final el manejo de los join´s del tipo de superposición.

Estas heurísticas priorizan los arcos/operaciones de selección (idénticos, subsunción y superposición) para ser procesados en el multi-grafo, dejando para el final los arcos de los join´s. A continuación se presenta el algoritmo de procesamiento de sub-expresiones comunes que utiliza estas heurísticas.

### 5.3.3 Algorithm Comsubproc:

**Input:** A multi-graph G'(R, SE, JE)

**Output:** A multi-graph G'(R', SE', JE') sin condiciones relacionadas con el mismo tipo arcos (SE or JE)

**Method:**

DO

Utilice las heurísticas para seleccionar un grupo de arcos/operaciones para procesar.

Identifique el tipo de arco/operación (p.e. selección) y lo común (p.e idéntica) entre las relaciones.

Realizar la operación en común o super-conjunto de este grupo de operaciones y modificar el multi-grafo, creando el nuevo nodo y eliminado los involucrados como se explicó en esta sección.

UNTIL no existen más operaciones en común.

En este algoritmo, cuando se elige un grupo de arcos este debe ser máximo, es decir, si hay tres operaciones idénticas se deben elegir las tres, no solo dos. También

es importante señalar que se pueden compartir las operaciones de reunión (join) antes de las operaciones de selección que no se comparten.

**Ejemplo 4: optimización de múltiples consultas**

A continuación se presenta un ejemplo que ayudara a entender los conceptos y pasos explicados anteriormente.

Supongamos que tenemos las siguientes relaciones c, o, e, co para los clientes, pedidos, empleados, y las empresas, respectivamente:

- c(cno, name, age)
- o(cno, date, item)
- e(name, employer, age, experience, salary, education)
- co(cname, location, earnings, president, business)

A continuación se presentan cuatro consultas para ser procesadas:

- Q9: $\Pi_{all}\, \sigma_{age} < 30$
- Q10: $\Pi_{all}\, \sigma_{age < 40 \wedge e.cno = c.cno}\, (e \times c)$
- Q11: $\Pi_{all}\, \sigma_{experience > 20 \wedge location = NY \wedge earnings}$ BETWEEN 300K AND 800K $\wedge$ $_{e.name = cname}\, (e \times co)$
- Q12: $\Pi_{all\ experience > 20 \wedge e.age < 65 \wedge location=NY \wedge\ earnings}$ BETWEEN 600K AND 900K $_{\wedge\ employer = cname\ AND\ e.name = president}\, (e \times co)$

Primero debemos descomponer este conjunto en dos grupos, Q9, Q10 y Q11, Q12, ya que estas se relacionan entre sí (no tendría sentido procesar las consultas Q9 y Q12 pues no tienen nada en común y no habrían beneficios). Ahora crearemos los multi-grafos para cada conjunto. La Figura 23(a) y 23(e) muestran los multi-grafos iniciales.

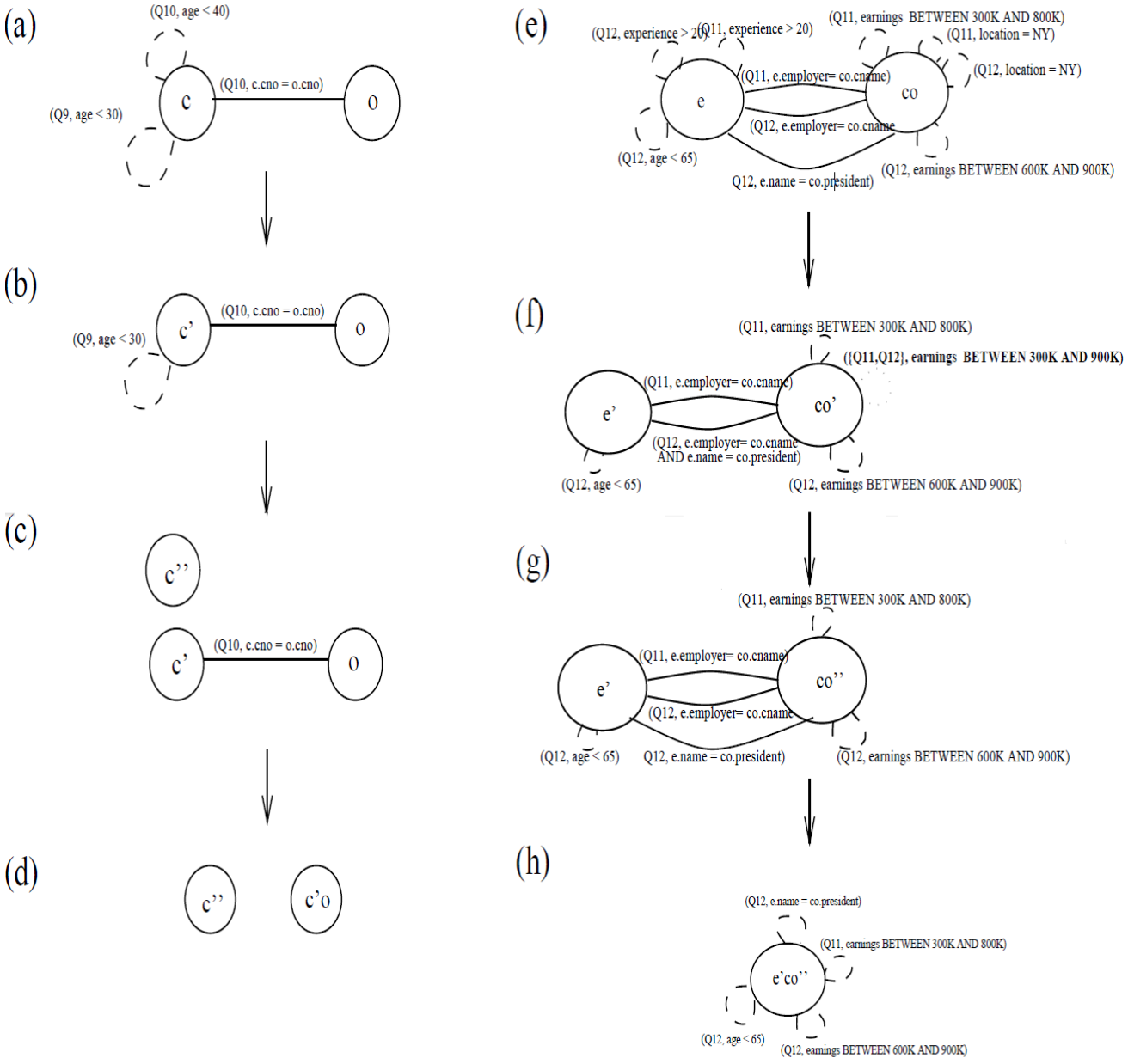



Figura 23: Ejemplo MPQ

En la Figura 23 (a) se ha aplicado la regla número 3 de las heurísticas encontrando la sub-expresión en común del tipo subsunción: (Q9, age < 30) and (Q10, age < 40). Luego se aplica el algoritmo (con una selección – subsunción) para procesar el predicado age < 40 en c. el resultado de esta operación en el multi-grafo se muestra en la Figura 23(b). En la Figura 23(b) no existen sub-expresiones comunes entre (Q9, age < 30). Por lo tanto podemos procesar (Q9, age < 30) y obtener el resultado c´´ en la Figura 23(c), para Q9. De forma similar, al realizar el join entre c´ y o (en la Figura 23 (d)), obtenemos el resultado para Q10.

Luego para el siguiente grupo de consultas en la Figura 23(e), usamos la regla número uno de las heurísticas para encontrar preposiciones idénticas de selección, (Q11, experience > 20), (Q12, experience > 20) y (Q11, location = NY), (Q12, location = NY). En la Figura 23(e) aún podemos encontrar más predicados en común en este caso de una superposición (overlap) entre dos operaciones de selección, por lo que se aplica la heurística número cuatro, (Q11, earnings BETWEEN 300K and 800K) y (Q12, earnings BETWEEN 600K and 900K). Por lo tanto se crea una restricción de super-conjunto, "earnings BETWEEN 300K and 900K" para Q11 y Q12. En la Figura 23(f) se encuentra el multi-grafo que muestra el resultado de las operaciones mencionadas. La Figura 23(g) muestra el resultado después de ejecutar el super-conjunto de la Figura 23(f). En la Figura 23(g), basados en la regla heurística número cinco, podemos obtener una ventaja gracias al predicado en común del join: e.employer=co.name. La Figura 23(h) muestra el multi-grafo resultante después de aplicar la operación en común del join. La condición original del join e.name=co.president de Q12 se convierte en un arco de selección del multi-grafo en la Figura 23(h).

## 5.4 Conclusiones

Los métodos de optimización actuales no garantizan encontrar el plan de ejecución más óptimo, pero si uno eficiente. Para mejorar el resultado de estos métodos de optimización es recomendable combinarlos.

Las características principales de la optimización, basada en heurísticas se pueden encontrar características semejantes en todos los otros métodos de optimización. Ya que estas características, satisfacen las necesidades de la optimización (reducir el tamaño de los resultados intermedios lo antes posible con operaciones de bajo costo), por ejemplo varios de los concejos de optimización del manual de MySQL en las sub-consultas.

Existen diferentes estudios respecto a la optimización de consultas múltiples sin embargo algunos no incluyen el caso de super-conjuntos para la superposición de consultas, quizás no en todos los casos sea conveniente considerarla, otros casos hacen referencias a bases de datos deductivas (Chakravarthy y J. Minker.) pero en este caso se estudia los casos relacionados a base de datos relacionales.

Al procesar el conjunto de consultas utilizando el algoritmo de MQP (expuesto en la Sección 5.5) puede que el resultado de una consulta en particular se atrase por el bien de la optimización global (ya que puede estar esperando un resultado de otra consulta), lo cual trae mayores beneficios.

Para lograr optimizar el procesamiento de múltiples consultas se podría modificar la forma actual de procesar una consulta (mencionada anteriormente en la Figura 20) llegando todas las consultas al optimizador del motor de base de datos. O

bien se podría realizar un pre-procesamiento antes de enviar cada consulta a la base de datos. De cualquier forma para obtener un mayor beneficio se puede combinar esta técnica con las vistas anteriormente, por ejemplo se podría utilizar las heurísticas de optimización con las consultas y luego el algoritmo para múltiples consultas o al mismo tiempo.

La optimización de múltiples consultas resulta muy útil, para las consultas de los reportes por la naturaleza de estas consultas, donde se encuentran varios cosas de igualdad en algunas operaciones, pero antes de aplicar la optimización de múltiples consultas se debe eliminar los errores que poseen y aplicar a estas consultas otros métodos de optimización (por ejemplo optimización semántica y optimización de sub-consultas según el manual de MySQL 5.0) para así obtener un resultado más eficiente.

# Capítulo VI: Desarrollo y Aplicación

## 6.1 Introducción

En este capítulo se aplican los métodos de optimización analizados en el Capítulo V, cuando corresponda y adaptándolos si es necesario, sobre las consultas de los reportes de estadística farmacia y monitoreo diario.

En primer lugar se realiza un cambio sintáctico, restructurando cada consulta y eliminando los predicados repetidos o innecesarios, con esto se da solución a algunos de los problemas detectados mencionados en la Sección 4.2, como por ejemplo la formula mal utilizada para contar cada receta y prescripción, obteniendo así que las consultas entreguen los resultados correctos.

Luego de tener las consultas correctamente estructuradas, se realiza la optimización de múltiples consultas, en conjunto con la optimización basada en heurísticas. Finalmente se presenta la implementación y resultados obtenidos gracias a esta optimización.

Las optimizaciones realizadas en este capítulo son centradas dentro del marco de MySQL 5.0.

## 6.2 Metodología para la optimización

En esta sección se da a conocer la forma de trabajo para esta optimización, primero para cada reporte se realizara una reformulación sintáctica de todas las consultas, donde se eliminan predicados que no aportan valores al resultados o que no son necesarios ya que producen errores. Luego de reformular todas las consultas con sus respectivas sub-consultas, se aplica la optimización basa en heurísticas y la optimización de múltiples consultas, pero el algoritmo y heurísticas de esta optimización son modificadas, por dos motivos: (1) la forma que en se comparten los resultados intermedios no es la más adecuada en términos de eficiencia. (2) poseemos información y datos estadísticos de la base de datos por lo que podemos sacarle el máximo provecho al algoritmo.

Dentro de la primera parte solo nos preocuparemos de predicados repetidos dentro de cada sub-consulta, sin considerar los predicados repetidos en las otras sub-consultas, de la misma consulta o de las otras consultas.

Luego de reformular todas las consultas y sub-consultas se aplica la segunda parte de esta optimización. Se decidió realizar la optimización heurística, en este punto ya que de esta forma bastará con aplicarlas a un menor número de consultas. Además, el manual de MySql recomienda realizar esta optimización pues "*el trabajo*

*en el optimizador de MySQL está en curso, por lo que esta sección está incompleta*." (Oracle and/or its affiliates, 2011)

Esta optimización es para un caso especificó por lo que la optimización para cada reporte, se realiza pre-procesando las consultas antes de ser enviadas al motor de la base de datos; y para representar y almacenar los valores intermedios (una acción en el multi-grafo) se crean nuevas tablas en la base de datos.

Por lo tanto se crean nuevas tablas en la base de datos según la necesidad de cada reporte. Con estas nuevas tablas se logra representaran los resultados intermedios de cada acción aplicada en el multi-grafo de MQP. A continuación se describe paso a paso como funciona esta optimización:

1. Al terminar de realizar la optimización de MQP y tener definidos los multi-grafos resultantes se sabe cuantas tablas se crean.
2. Cuando se ejecute o sea llamado el reporte estas tablas son llenadas con la información requerida, según las consultas y el multi-grafo.
3. Al terminar la ejecución del reporte los datos de estas tablas son eliminados.
4. Como se insertan y se eliminan los datos de las nuevas tablas cada vez que es llamado el reporte se manejan las tablas de tal forma de no eliminar los datos que otro usuario podría estar utilizando al mismo tiempo.
5. El crear tablas e insertar datos, también tienen un costo. Cuando se habla de costos en este caso y para el resto del capítulo, se considera el costo de mantener nuevas tablas en la base de datos (almacenamiento), el costo de insertar tuplas por medio de una sentencia insert select y el costo de eliminar los datos.

Para este caso se utiliza el multi-grafo definido por MQP en la Sección 5.3.1 pero las heurísticas que indican en que orden realizar las acciones sobre el multi-grafo son modificadas. Esta modificación se realizo por los siguientes motivos:

- Se deben crear el menor número de tablas para mantener los resultados temporales por su alto costo.
- Se posee información estadística de la bases de datos y del sistema. Por lo que se puede obtener mejores resultados.

Para el uso de estas nuevas heurísticas se hace uso de una tabla de predicados que contendrá todos los predicados del conjunto de consultas que se desean ejecutar, indicando: el predicado, el tipo al que corresponde (selección idéntica, selección con subsunción, etc.), el tipo de consulta (esto es equivalente al identificador de la

consulta pero de forma global), el factor del número de consultas involucradas (Fc) y el factor de selectividad del predicado (Fs). Un ejemplo de esta tabla se ilustra en la Tabla 4.

| Predicado | Tipo | Tipo consulta | Fc | Fs |
|---|---|---|---|---|
| detalle_tarjeta_farmacia.entregado=1 | Igualdad | Todas | 1 | 1 |
| receta_farmacia.id_clasificacion = 1 | Igualdad | Grupo etario 1 | 0,243 | 0,66 |
| detalle_tarjeta_farmacia inner join paciente on paciente | Join igualdad | Sectores | 0,756 | |

Tabla 4: Tabla de Predicados.

Donde Fc, representa el número consultas que utilizan el predicado, dividido por el número total de consultas en el conjunto. Por ejemplo si el predicado $p_x$ lo utilizan 10 consultas de un universo de 20 consultas su Fc es igual a 0.5. En este caso indica que el 50% de las consultas utiliza el predicado, un valor cercano a 1 tiene una mayor prioridad pues más consultas son beneficiadas.

Fs, corresponde al factor de selectividad del predicado. Este corresponde a la cardinalidad de ejecución del predicado, dividido por la cardinalidad total de la tabla a la que pertenece. Por ejemplo si el predicado $p_x$ al ser ejecutado retorna 200 tuplas y la cardinalidad total de la tabla corresponde a 300 tuplas su Fs será igual a 0,66 esto indica que se espera que su valor retorne un 66% de las tuplas pertenecientes a la tabla. En este caso interesan los valores de Fs cercanos a 0.5. Ya que un valor cercano a 1 indica crear un resultado intermedio tan grande como la tabla original y para un valor cercano a 0 no valdría la pena crear una nueva tabla. Además este valor es utilizado para romper empates en predicados del mismo tipo y de igual Fc.

Se ha establecido de antemano con el encargado de informática del CESFAM Violeta Parra que crear más de 4 tablas (por reporte) para mantener los resultados temporales no es recomendable, al igual que manejar resultados temporales con una selectividad menor al 10% o mayor al 90% por los motivos ya mencionados.

Quedando las heurísticas de MQP de la siguiente manera:

1. Ejecutar todos los predicados de selección idénticos con Fc=1. Luego de este punto cualquier acción en el multi-grafo nos hará crear al menos un nodo nuevo (lo que significara crear tablas para mantener los resultados temporales).

2. Ejecutar los predicados de selección idénticos con el mayor Fc, si el Fs es menor a 01 o mayor a 0.9 ignorar este predicado y estudiar el siguiente predicado.
3. Ejecutar los predicados de selección del tipo subsunción cuyo Fc es mayor ejecutando una después de otra (como se explicó anteriormente), si su Fs es menor a 01 o mayor 0.9 considerar crear un super-conjunto o ignorar este predicado y evaluar el siguiente predicado de selección del tipo subsunción.
4. Crear super-conjuntos para los predicados del tipo de superposición que poseen el mayo Fc si el Fs es menor a 01 o mayor a 0.9 ignorar este predicado y estudiar el siguiente predicado.
5. Ejecutar los predicados de join del tipo de igualdad con el mayor Fc, dejando para el final los otros predicados de join.
6. Para cualquiera de los puntos anteriores. Si existen dos o más predicados con igual Fc utilizar el que posea un Fs más cercano al valor de 0.5. y luego evaluar el los predicados rechazados.
7. Se recomienda finalizar la transformación del multi-grafo cuando se hayan creado un valor igual o menor a 4 tablas para mantener los resultados temporales y ejecutar el resto de los predicados sobre estos resultados temporales de forma normal.

Como se puede observar la optimización de MQP no se ejecuta de forma completa pues el costo de crear, insertar y eliminar cada tabla es alto.

### 6.2.1 Reporte Estadística Farmacia

Primero se analiza la consulta que genera el total de las recetas en el resumen de estadística farmacia (Figura 24, cuadro en rojo).

| RESUMEN | 0-9 AÑOS | | 10-19 AÑOS | | 20-64 AÑOS | | 65 AÑOS Y MÁS | | TOTAL | |
|---|---|---|---|---|---|---|---|---|---|---|
| | R | P | R | P | R | P | R | P | R | P |
| Morbilidad Diurno | 712 | 1386 | 302 | 559 | 1837 | 3546 | 446 | 968 | 3297 | 6459 |
| Morbilidad Vespertino | 59 | 122 | 111 | 197 | 785 | 1797 | 179 | 405 | 1134 | 2521 |
| **Total Morbilidad** | 771 | 1508 | 413 | 756 | 2622 | 5343 | 625 | 1373 | 4431 | 8980 |
| Morbilidad Diurno No | 0 | 0 | 0 | 0 | 4 | 10 | 1 | 0 | 5 | 10 |
| Morbilidad Vespertino No | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Total Morbilidad No** | 0 | 0 | 0 | 0 | 4 | 10 | 1 | 0 | 5 | 10 |
| SAPU Diurno | 97 | 242 | 34 | 95 | 228 | 676 | 59 | 157 | 418 | 1170 |
| SAPU Vespertino | 269 | 634 | 198 | 491 | 836 | 2488 | 117 | 361 | 1420 | 3974 |
| **Total SAPU** | 366 | 876 | 232 | 586 | 1064 | 3164 | 176 | 518 | 1838 | 5144 |
| SAPU Diurno No Desp. | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| SAPU Vespertino | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| **Total SAPU No Desp.** | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| Crónico Diurno | 141 | 245 | 143 | 255 | 5457 | 17045 | 5040 | 20853 | 10781 | 38398 |
| Crónico Vespertino | 10 | 20 | 12 | 33 | 935 | 2904 | 834 | 3499 | 1791 | 6456 |
| **Total Crónico** | 151 | 265 | 155 | 288 | 6392 | 19949 | 5874 | 24352 | 12572 | 44854 |
| **Total Despachado** | 1288 | 2649 | 800 | 1630 | 10078 | 28456 | 6675 | 26243 | 18841 | 58978 |
| **Total No Despachado** | 0 | 0 | 0 | 0 | 5 | 10 | 1 | 0 | 6 | 10 |

Figura 24: Consulta, Resumen Estadística Farmacia.

Dentro de esta consulta se ejecutan cuatro sub-consultas que no son utilizadas, es decir, se ejecutan pero sus resultados no son utilizados, estos valores son: (1) receta crónico vespertino no despachado, (2) receta crónico diurno no despachado, (3) prescripción crónico vespertino no despachado y (4) prescripción crónico diurno no despachado. Estas no son consideradas ya que no fueron solicitadas en los requerimientos sin embargo se crearon por si en el futuro había un cambio en los requerimientos. Estas sub-consultas están presentes dentro de todas las consultas, esto significa que cada vez que se solicita este reporte se ejecutan 180 sub-consultas innecesarias por lo que estas deben ser eliminadas o dejar como comentario.

En la Figura 25 se ilustra la sub-consulta que genera las prescripciones de pacientes diurnos de procedencia morbilidad de la sección del resumen y en ella se detallan los problemas ya detectados. Los siguientes cambios son aplicados a esta y a todas las sub-consultas del reporte estadística farmacia.

La primera modificación corresponde a cambiar la formula que se utiliza para contar el número de recetas y prescripciones, como se menciono anteriormente esta formula esta bien definida solo para los pacientes de procedencia crónica. Quedando la formula para calcular las recetas:

```
[...] (select COUNT(DISTINCT(folio_receta_farmacia)) as diurno_Morbilidad_receta [...]
```

Y la formula para calcular las prescripciones:

```
[...] (select COUNT(folio_receta_farmacia) as diurno_Morbilidad [...]
```

Luego se puede observar que para obtener el resultado en esta sección no es necesario utilizar la tabla paciente. Esta consulta esta generando un join con la tabla paciente pero no utiliza ningún atributo de esta. Y para eliminar los predicados repetidos dentro de cada sub-consulta utilizaremos la clausula where para ordenar los predicados (observar que el horario de atención del día sábado es igual al del día domingo). Quedando así la clausula from solo con el join necesario. También aprovecharemos de aplicar algunos concejos de MySQL (Oracle and/or its affiliates, 2011) respecto a las sub-consultas, por ejemplo remplazando:

```
        and dayname(detalle_tarjeta_farmacia.fecha_entrega)<>'Friday'
        and dayname(detalle_tarjeta_farmacia.fecha_entrega)<>'Saturday'
        and dayname(detalle_tarjeta_farmacia.fecha_entrega)<>'Sunday'

    and dayname(detalle_tarjeta_farmacia.fecha_entrega)not in ('Friday', 'Saturday', 'Sunday'))
```

→

Este cambio también se aplicó en las consultas que generan las recetas y prescripciones sin sector ya que se preguntaba de la misma forma si era distinto del sector 1, distinto del sector 2, distinto del sector 3, distinto del sector 4, distinto del

sector 5 y distinto del sector 6. Dando como resultado una consulta estructurada como es recomendado por el manual de MySQL, esta se ilustra en la Figura 26.


Figura 25: Sub-consulta Prescripciones Diurno Morbilidad.

```
(select COUNT(folio_receta_farmacia) as diurno_Morbilidad
from
    detalle_tarjeta_farmacia inner join receta_farmacia on
    receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
where
    detalle_tarjeta_farmacia.procedencia='Morbilidad'
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and detalle_tarjeta_farmacia.entregado=1
    and detalle_tarjeta_farmacia.fecha_entrega between '2011-07-01' and '2011-07-31'
    and ((detalle_tarjeta_farmacia.hora<'17:00:00'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)not in ('Friday', 'Saturday', 'Sunday'))
    or (detalle_tarjeta_farmacia.hora<'16:00:00'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)='Friday')
    or (detalle_tarjeta_farmacia.hora<'9:00:00'
    and (dayname(detalle_tarjeta_farmacia.fecha_entrega)='Saturday'
    or dayname(detalle_tarjeta_farmacia.fecha_entrega)='Sunday'))
    ) ) as diurno_Morbilidad,
```

Figura 26: Sub-consulta Resultante Prescripciones Diurno Morbilidad.

Hasta el momento solo se ha realizado un cambio sintáctico aplicando algunos concejos de optimización semántica y del manual de MySQL. Estos cambios y otros se han realizado para todas las consultas con sus respectivas sub-consultas de este reporte. Considere que las consultas que utilizan el sector si utilizan la tabla paciente. Consiguiendo así los resultados deseados y eliminar predicados repetidos (se han eliminado 15 líneas de código aproximadamente por sub-consulta).

El siguiente paso para esta optimización, es realizar la optimización de MQP en conjunto con la optimización basada en heurísticas, en donde se consideraran todas las sub-consultas y consultas de este reporte.

El primer paso es identificar los puntos en común de las consultas y sub-consultas. A continuación se presentan los puntos en común de las consultas y sub-consultas en la Tabla 5. Descrita anteriormente.

| Predicado | Tipo | Tipo consulta | Fc | Fs |
|---|---|---|---|---|
| detalle_tarjeta_farmacia.entregado=1 | Igualdad | Todas | 1 | 1 |
| receta_farmacia inner join detalle_tarjeta_farmacia | Join igualdad | Todas | 1 | |
| detalle_tarjeta_farmacia inner join paciente on paciente | Join igualdad | Sectores | 0,756 | |
| detalle_tarjeta_farmacia.cantidad_entregada>0 | Igualdad | Despachas | 0,6 | 0,994 |

| Predicado | Tipo | Tipo consulta | Fc | Fs |
|---|---|---|---|---|
| detalle_tarjeta_farmacia.fecha_entrega between '2011-07-01' and '2011-07-31' | Igualdad | Prescripción | 0,5 | 0,170 |
| cast(receta_farmacia.fecha as date) between '2011-07-01' and '2011-07-31' | Igualdad | Receta | 0,5 | 0,168 |
| detalle_tarjeta_farmacia.cantidad_entregada=0 | Igualdad | No despachadas | 0,4 | 0,005 |
| detalle_tarjeta_farmacia.procedencia= 'Morbilidad' | Igualdad | Morbilidad | 0,4 | 0,254 |
| detalle_tarjeta_farmacia.procedencia= 'SAPU' | Igualdad | SAPU | 0,4 | 0,144 |
| receta_farmacia.id_clasificacion = 1 | Igualdad | Grupo etario 1 | 0,243 | 0,107 |
| receta_farmacia.id_clasificacion = 2 | Igualdad | Grupo etario 2 | 0,243 | 0,064 |
| receta_farmacia.id_clasificacion = 3 | Igualdad | Grupo etario 3 | 0,243 | 0,545 |
| receta_farmacia.id_clasificacion = 4 | Igualdad | Grupo etario 4 | 0,243 | 0,283 |
| detalle_tarjeta_farmacia.procedencia= 'Crónico' | Igualdad | Crónico | 0,2 | 0,601 |
| detalle_tarjeta_farmacia inner join producto_generico | Join igualdad | No despachadas | 0,2 | |
| producto_generico.arsenal=1 | Igualdad | No despachadas | 0,2 | 0,018 |
| detalle_tarjeta_farmacia.hora>='17:00:00' and dayname(detalle_tarjeta_farmacia.fecha_entrega)not in ('Friday', 'Saturday', 'Sunday')) or (detalle_tarjeta_farmacia.hora>='16:00:00'and dayname(detalle_tarjeta_farmacia.fecha_entrega)='Friday') or (detalle_tarjeta_farmacia.hora>='9:00:00' and (dayname(detalle_tarjeta_farmacia.fecha_entrega)='Saturday' or dayname(detalle_tarjeta_farmacia.fecha_entrega)='Sunday') | Igualdad | Prescripción vespertina | 0,2 | 0,288 |

| Predicado | Tipo | Tipo consulta | Fc | Fs |
|---|---|---|---|---|
| detalle_tarjeta_farmacia.hora<'17:00:00' and dayname(detalle_tarjeta_farmacia.fecha_entrega)not in ('Friday', 'Saturday', 'Sunday')) or (detalle_tarjeta_farmacia.hora<'16:00:00'and dayname(detalle_tarjeta_farmacia.fecha_entrega)='Friday') or (detalle_tarjeta_farmacia.hora<'9:00:00' and (dayname(detalle_tarjeta_farmacia.fecha_entrega)='Saturday' or dayname(detalle_tarjeta_farmacia.fecha_entrega)='Sunday') | Igualdad | Prescripción diurna | 0,2 | 0,711 |
| (cast(receta_farmacia.fecha as time)>='17:00:00' and dayname(cast(receta_farmacia.fecha as date))not in ('Friday', 'Saturday', 'Sunday')) or (cast(receta_farmacia.fecha as time)>='16:00:00' and dayname(cast(receta_farmacia.fecha as date))='Friday') or (cast(receta_farmacia.fecha as time)>='9:00:00' and (dayname(cast(receta_farmacia.fecha as date))='Saturday' or dayname(cast(receta_farmacia.fecha as date))='Sunday') | Igualdad | Receta vespertina | 0,2 | 0,223 |
| (cast(receta_farmacia.fecha as time)<'17:00:00' and dayname(cast(receta_farmacia.fecha as date))not in ('Friday', 'Saturday', 'Sunday')) or (cast(receta_farmacia.fecha as time)<'16:00:00' and dayname(cast(receta_farmacia.fecha as date))='Friday') or (cast(receta_farmacia.fecha as time)<'9:00:00' and (dayname(cast(receta_farmacia.fecha as date))='Saturday' or dayname(cast(receta_farmacia.fecha as date))='Sunday') | Igualdad | Receta diurna | 0,2 | 0,776 |
| substr(paciente.numFamilia, 2, 1)= '1' | Igualdad | Sector 1 | 0,108 | 0,145 |
| substr(paciente.numFamilia, 2, 1)= '2' | Igualdad | Sector 2 | 0,108 | 0,194 |
| substr(paciente.numFamilia, 2, 1)= '3' | Igualdad | Sector 3 | 0,108 | 0,103 |
| substr(paciente.numFamilia, 2, 1)= '4' | Igualdad | Sector 4 | 0,108 | 0,142 |
| substr(paciente.numFamilia, 2, 1)= '5' | Igualdad | Sector 5 | 0,108 | 0,122 |

| Predicado | Tipo | Tipo consulta | Fc | Fs |
|---|---|---|---|---|
| substr(paciente.numFamilia, 2, 1)= '6' | Igualdad | Sector 6 | 0,108 | 0,157 |
| substr(paciente.numFamilia, 2, 1)not IN ('1', '2', '3', '4' ,'5', '6') | Igualdad | Sin sector | 0,108 | 0,135 |

Tabla 5: Predicados Comunes del Reporte de Estadística Farmacia.

Como se puede observar existen muchos predicados en común (27 predicados) y todos son del tipo de igualdad. Algunas afectan a más consultas o sub-consultas que otras.

Para no ir sobre cargando el grafo se agregan los arcos de condición mediante se ejecutan los predicados según las heurísticas establecidas. Además se añade una explicación correspondiente. En la Figura 27 (a) se presenta el primer grafo creado.

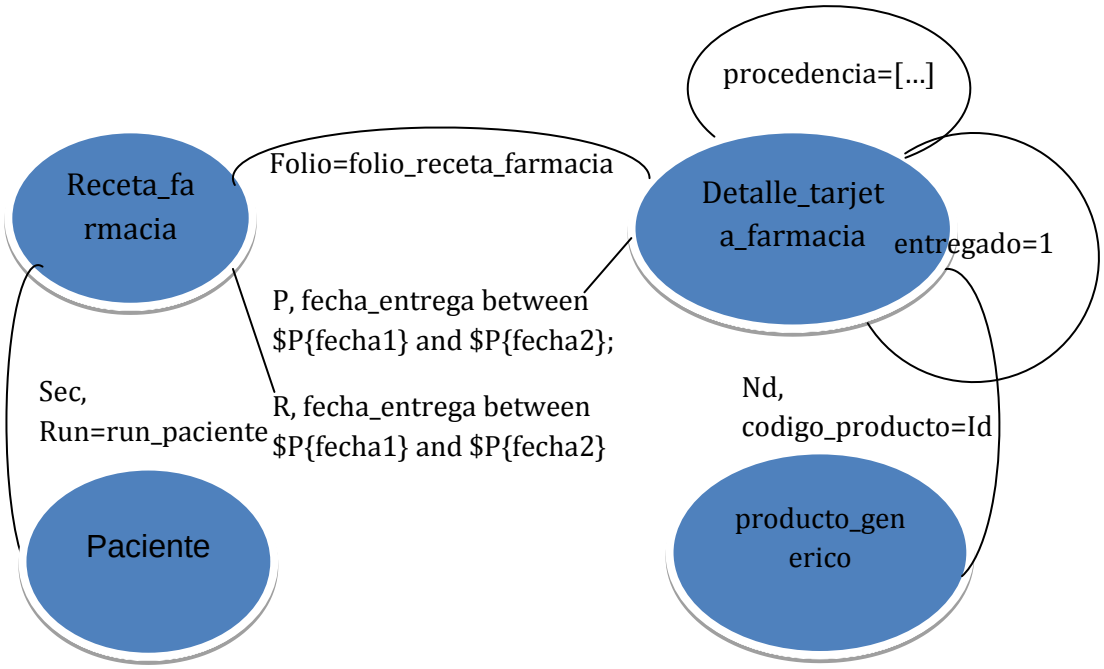



Figura 27 (a): Multi-grafo Inicial

Siguiendo las heurísticas la primera acción a realizar es el predicado "entregado=1" que afecta a todas las consultas y sub-consultas. Quedando el multigrafo como se ilustra en la Figura 27 (b).

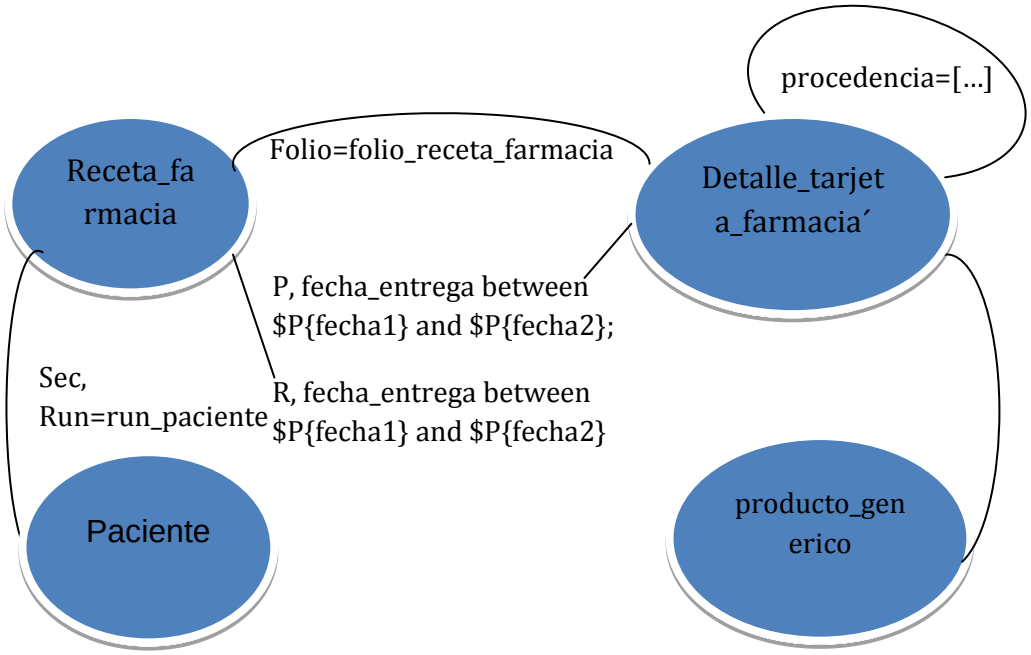



Figura 27 (b): Multi-grafo Primera Acción

Ahora utilizaremos la segunda heurística y el siguiente predicado en nuestra tabla es cantidad_entregada>0 pero su Fs es igual 0,994 por lo que será descartado.

Los siguientes predicados corresponden a los de tipo fecha pero no podemos ejecutar estos predicados por separado ya que si se utiliza puede darse el caso que para una fecha dada una receta no entre dentro del rango pero si su prescripción (por ser retirada después), pero al momento de realizar el join no se encontraría la clave foránea. Por lo que esta acción se realizara después de realizar el join de receta_farmacia con detalle_targeta_farmacia creando un super-conjunto.

El predicado cantidad_entregada=0 corresponde al siguiente predicado con mayor Fc pero tampoco cumple con los requisitos de las heurísticas por su bajo Fs.

Los predicados procedencia= 'Morbilidad' y de procedencia= 'SAPU' cumplen con las heurísticas por lo que serán ejecutado en el multi-grafo pero podemos ver que son excluyentes y al realizar el predicado de procedencia= 'Crónico' englobamos a todas las consultas. A demás el predicado de procedencia= 'Crónico' posee un factor de Fs muy cercano al valor esperado. Al ejecutar estos predicados se dividirá el multi-grafo en tres multi-grafos distintos como se ilustra en las Figuras 28 (a), 28 (b), 28 (c).

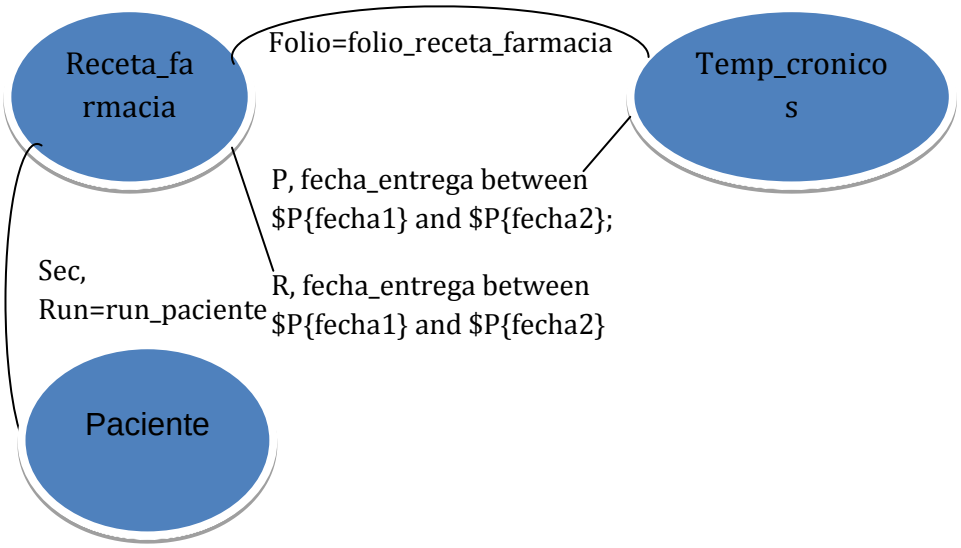



Figura 28 (a): Multi-grafo Segunda Iteración.

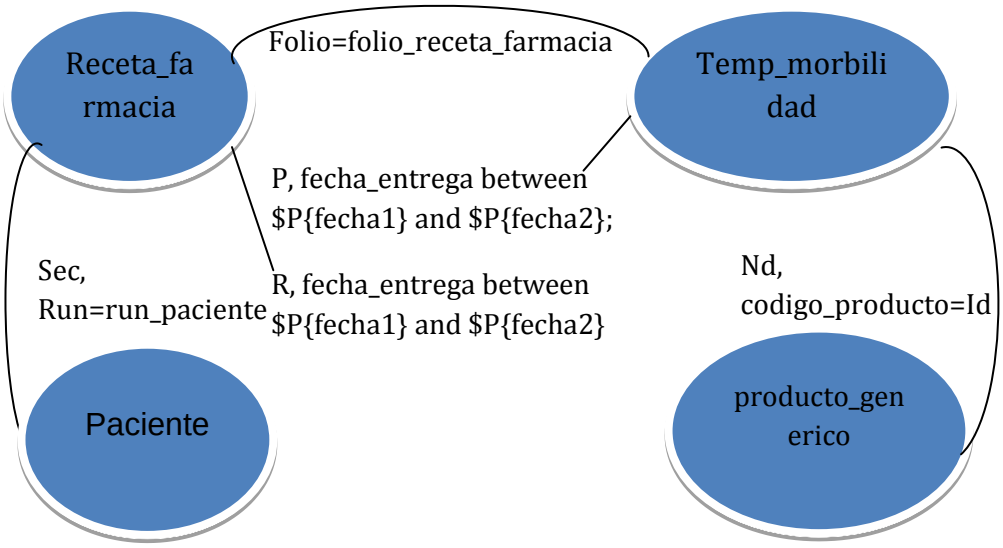



Figura 28 (b): Multi-grafo Segunda Iteración.

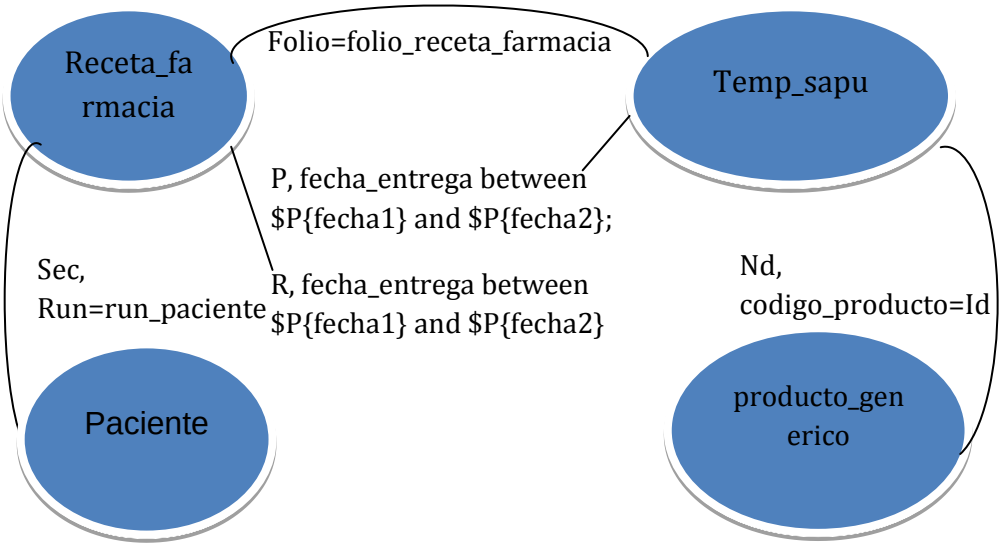



Figura 28 (c): Multi-grafo Segunda Iteracion.

Como se puede observar el multi-grafo de los pacientes crónicos no posee el join con producto_generico ya que en este caso no se consideran productos no despachados. Si se desea seguir iterando los siguientes predicados deberán ejecutarse sobre cada uno de estos múltiples-grafos.

El siguiente paso es seleccionar la siguiente acción a realizar sobre los multi-grafos existentes. Los siguientes predicados corresponden a los predicados de clasificación sin embargo estos no se consideraran por la estructura del reporte (el reporte utiliza estos datos para iterar). Además una siguiente iteración con la aplicación de cualquier predicado nos haría pasar de 3 tablas temporales a 6. Por lo que las siguientes acciones a realizar corresponden a los predicados de join.

Por lo tanto la siguiente acción correspondería a un arco de join.

- El primer join que se recomienda según la Tabla 5 es el join con receta_farmacia, ese es realizado por todas las consultas y sub-consultas, por lo que si será ejecutado en este punto.
- Ya que realizaremos este join aprovecharemos de crear un super-conjunto para “fecha_entrega between $P{fecha1} and $P{fecha2}” y “cast(receta_farmacia.fecha as date) between $P{fecha} and $P{fecha}” que correspondería a: (fecha_entrega between $P{fecha1} and $P{fecha2} or cast(receta_farmacia.fecha as date) between $P{fecha} and $P{fecha}). Como se menciono anteriormente.
- Además, para evitar otra iteración y la creación de más tablas, ejecutaremos en este punto el siguiente join en la tabla, correspondiente al join con paciente.

Ejecutar esta acción es conveniente por los siguientes motivos: (1) este afecta un gran número de consultas y sub-consultas. (2) de paciente solo necesitamos un solo atributo (numFamilia) y el tamaño de la nueva tabla solo depende de la tabla detalle_tarjeta_farmacia. (3) crear nuevas tablas tiene un alto costo por lo que debemos intentar de crear la menor cantidad "de resultados intermedios".

- La justificación anterior también se podría aplicar para el join de producto_generico pero este predicado solo afecta 4 sub-consultas por consulta. Además el predicado que necesita este join tiene un factor de selectividad cercano a cero.

Los multi-grafos que se ilustran en las Figuras 29 (a), 29 (b) y 29 (c) son los multi-grafos resultantes para los multi-grafos anteriores.

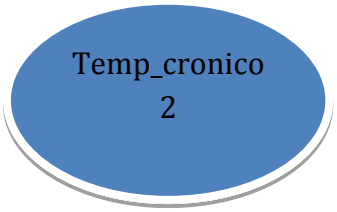



Figura 29 (a): Multi-grafo Resultante para Procedencia Crónica.

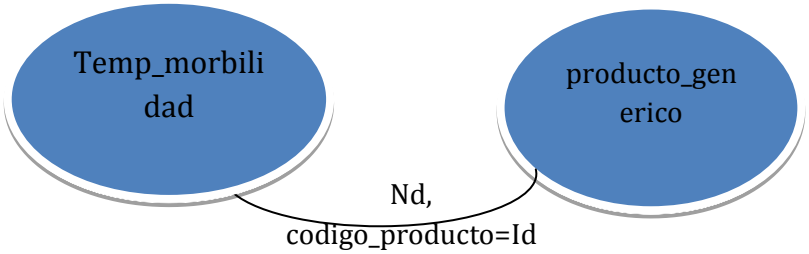



Figura 29 (b): Multi-grafo Resultante para Morbilidad.

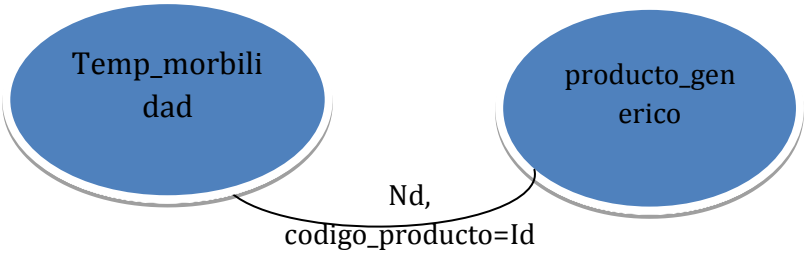



Figura 29 (c): Multi-grafo Resultante para SAPU.

Al momento de crear las tablas que mantendrán los resultados de estas operaciones se aprovechara de seleccionar solo los atributos necesarios para las

siguientes consultas que se efectuaran sobre cada tabla (o nodo). Esto seria equivalente a realizar optimización basada en heurísticas (ejecutar selección y proyección lo antes posible).

El siguiente paso es modificar las consultas y sub-consultas existentes, que trabajaran sobre estos resultados temporales, por ejemplo para la sub-consulta del sector uno que genera las prescripciones de los pacientes de procedencia de morbilidad en horario diurno, que se presenta en la Figura 30 (a), debemos modificar la clausula del from pues ya no es necesario realizar ningún join y vasta con preguntar sobre la nueva tabla y tampoco es necesario preguntar por el predicado "entregado=1" y por el predicado de procedencia = "Morbilidad", esta sub-consulta modificada se muestra en la Figura 30 (b).

```
(select COUNT(folio_receta_farmacia) as diurno_Morbilidad
from
    detalle_tarjeta_farmacia inner join receta_farmacia on
    receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    inner join paciente on paciente.run = receta_farmacia.run_paciente
where
    detalle_tarjeta_farmacia.procedencia='Morbilidad'
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and detalle_tarjeta_farmacia.entregado=1
    and detalle_tarjeta_farmacia.fecha_entrega between '2011-07-01' and '2011-07-31'
    and receta_farmacia.id_clasificacion = 1
    and  substr(paciente.numFamilia, 2, 1)= '1'
    and ((detalle_tarjeta_farmacia.hora<'17:00:00'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)not in ('Friday', 'Saturday', 'Sunday'))
    or (detalle_tarjeta_farmacia.hora<'16:00:00'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)='Friday')
    or (detalle_tarjeta_farmacia.hora<'9:00:00'
    and (dayname(detalle_tarjeta_farmacia.fecha_entrega)='Saturday'
    or dayname(detalle_tarjeta_farmacia.fecha_entrega)='Sunday'))
    ) ) as diurno_Morbilidad,
```

Figura 30 (a): Sub-consulta Prescripciones Diurno Morbilidad

```
(select COUNT(folio) as diurno_Morbilidad
from
    temp_morbilidad
where
    temp_morbilidad.cantidad_entregada>0
    and temp_morbilidad.fecha_entrega between $P{fecha1} and $P{fecha2}
    and temp_morbilidad.id_clasificacion = $P{clas_id}
    and substr(temp_morbilidad.numFamilia, 2, 1)= '1'
    and ((temp_morbilidad.hora<'17:00:00'
    and dayname(temp_morbilidad.fecha_entrega)not in ('Friday', 'Saturday', 'Sunday'))
    or (temp_morbilidad.hora<'16:00:00'
    and dayname(temp_morbilidad.fecha_entrega)='Friday')
    or (temp_morbilidad.hora<'9:00:00'
    and (dayname(temp_morbilidad.fecha_entrega)='Saturday'
    or dayname(temp_morbilidad.fecha_entrega)='Sunday'))
    ) ) as diurno Morbilidad,
```

Figura 30 (b): Nueva Sub-consulta Prescripciones Diurno Morbilidad.

Estos cambios fueron realizados para todas las consultas y sub-consultas de este reporte y para aquellas sub-consultas que hacían uso de los predicados de "detalle_tarjeta_farmacia inner join producto_generico" y "producto_generico.arsenal=1" se realizo la siguiente modificación utilizando la nueva tabla y la optimización basada en heurísticas, envés de hacer los join correspondientes primero seleccionamos los productos de que pertenecen al arsenal 1 de la tabla producto_generico pues como se menciono anteriormente hasta ahora solo 10 productos cumplen con esta condición, para ilustrar esta acción se muestra la Figura 31(a) la sub-consulta que generaba las prescripciones SAPU no despachadas en horario vespertino y en la Figura 31 (b) la sub-consulta nueva e equivalente.

```
(select COUNT(folio_receta_farmacia) as vespertino_SAPU_no_desp
from
    detalle_tarjeta_farmacia inner join receta_farmacia on
    receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    inner join paciente on paciente.run = receta_farmacia.run_paciente
    inner join producto_generico on detalle_tarjeta_farmacia.codigo_producto= producto_generico.Id
where
    detalle_tarjeta_farmacia.procedencia='SAPU'
    and receta_farmacia.id_clasificacion = 1
    and  substr(paciente.numFamilia, 2, 1)= '1'
    and  detalle_tarjeta_farmacia.cantidad_entregada=0
    and detalle_tarjeta_farmacia.entregado=1
    and  producto_generico.arsenal=1
    and detalle_tarjeta_farmacia.fecha_entrega between '2011-07-01' and '2011-07-31'
    and detalle_tarjeta_farmacia.hora>='18:00:00')  as vespertino_SAPU_no_desp,
```

Figura 31 (a): Sub-consulta Prescripciones SAPU no Despachadas, Horario Vespertino.

```
(select COUNT(folio) as vespertino_SAPU_no_desp
from
    (select id from producto_generico where arsenal =1) as productosA
    inner join temp_sapu on productosA.id = temp_sapu.codigo_producto
where
    temp_sapu.id_clasificacion = $P{clas_id}
    and substr(temp_sapu.numFamilia, 2, 1)= '1'
    and temp_sapu.cantidad_entregada=0
    and temp_sapu.fecha_entrega between $P{fecha1} and $P{fecha2}
    and temp_sapu.hora>='18:00:00')  as vespertino_SAPU_no_desp,
```

Figura 31 (b): Nueva sub-consulta Prescripciones SAPU no Despachadas, Horario Vespertino.

### 6.2.2 Reporte Monitoreo Diario

Al igual que en el reporte de estadística farmacia se reformulan las consultas. A continuación se analiza la consulta que genera la primera sección: total de recetas y prescripciones, como se ilustra en el Figura 32.

Para generar RC (recetas de los pacientes crónicos) se ejecutan dos sub-consultas por día, una sub-consulta genera las recetas de crónicas en horario diurno y la otra en horario vespertino (luego estas se suman). En la Figura 33 se ilustra la sub-consulta que genera las recetas de los pacientes crónico diurno. Como podemos observar esta sub-consulta es exactamente igual a la de reporte estadística farmacia (al igual que todas las que calculan las recetas) que calcula el mismo valor en la sección del resumen. Por lo tanto, se presentan los mismos errores de repetición de predicados y se realizaran los mismos cambios.

MES: 6
AÑO: 2011

| DÍA | N | | | |
|---|---|---|---|---|
| | RC | RM | PC | PM |
| 1 | 582 | 267 | 2052 | 574 |
| 2 | 601 | 250 | 2301 | 539 |
| 3 | 483 | 275 | 1789 | 690 |
| 4 | 30 | 119 | 103 | 275 |
| 5 | 0 | 105 | 0 | 329 |

Figura 32: Primera Sección.

En esta sección se podría pensar en generar las recetas sin considerar las clausulas para los horarios diurnos y vespertinos, ya que se necesita el total de las recetas, pero existe la posibilidad de que prescripciones de una misma receta se

retiren en horario diurno y vespertino, al no haber en ese momento (horario diurno) disponibilidad de productos o por errores. Por lo que se debe contar como otra receta.

Para calcular las prescripciones se utiliza la consulta de la Figura 34. Como en este caso no son necesarios los datos de clasificación o grupo etario no es necesario ejecutar ningún join y solo se trabaja con la tabla detalle_tarjeta_farmacia. Para obtener el valor total de prescripciones de morbilidad se debe calcular las prescripciones de SAPU y morbilidad (como se había mencionado anteriormente).

Los primeros cambios de este reporte (al igual que en el reporte de farmacia estadística) es cambiar la formula para calcular el número de recetas y prescripciones para SAPU y morbilidad (recordar que esta formula esta bien aplicada para los pacientes de procedencia crónica). Para las sub-consultas que calculan las recetas se realizaran los mismos cambios, que los realizados en el reporte de estadística farmacia y para las prescripciones se modificara la sub-consulta que calcula las prescripciones de morbilidad, preguntando por la procedencia de SAPU y morbilidad al mismo tiempo, es decir, envés de realizar dos sub-consultas y luego sumarlas se modificara el predicado de la siguiente forma:

```
and detalle_tarjeta_farmacia.procedencia= 'Morbilidad'
    and (detalle_tarjeta_farmacia.procedencia= 'SAPU'
→   or detalle_tarjeta_farmacia.procedencia= 'Morbilidad')) y se eliminara
```

una sub-consulta.

Recordar que estas sub-consultas se iteran por cada día del mes seleccionado.

```
(select IFNULL(sum( if(
convert(DATEDIFF (proxEnt, ent)/30, signed int)=0, 1, convert(DATEDIFF (proxEnt, ent)/30, signed int) )),0) as receta
from
    (select distinct receta_farmacia.folio as rec,
        (select max(detalle_tarjeta_farmacia.fecha_prox_entrega)
        from detalle_tarjeta_farmacia
        where detalle_tarjeta_farmacia.folio_receta_farmacia= rec) as proxEnt,
        receta_farmacia.fecha as ent
 from
    receta_farmacia inner join detalle_tarjeta_farmacia on
    receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and detalle_tarjeta_farmacia.entregado=1
    and cast(receta_farmacia.fecha as date)=$P{fecha}
    and detalle_tarjeta_farmacia.procedencia= 'Crónico'
    and cast(receta_farmacia.fecha as time)<'17:00:00'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)<>'Friday'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)<>'Saturday'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)<>'Sunday'

    or receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    and detalle_tarjeta_farmacia.procedencia='Crónico'
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and cast(receta_farmacia.fecha as date)=$P{fecha}
    and detalle_tarjeta_farmacia.entregado=1
    and cast(receta_farmacia.fecha as time)<'16:00:00'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)='Friday'

    or receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    and detalle_tarjeta_farmacia.entregado=1
    and detalle_tarjeta_farmacia.procedencia='Crónico'
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and cast(receta_farmacia.fecha as date)=$P{fecha}
    and cast(receta_farmacia.fecha as time)<'9:00:00'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)='Saturday'

    or receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    and detalle_tarjeta_farmacia.entregado=1
    and detalle_tarjeta_farmacia.procedencia='Crónico'
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and cast(receta_farmacia.fecha as date)=$P{fecha}
    and cast(receta_farmacia.fecha as time)<'9:00:00'
    and dayname(detalle_tarjeta_farmacia.fecha_entrega)='Sunday' ) as tab) as receta_cronico_diurno,
```

Figura 33: Sub-consulta Recetas Crónico Diurno.

```
(select IFNULL(sum(if( convert (DATEDIFF (detalle_tarjeta_farmacia.fecha_prox_entrega, detalle_tarjeta_farmacia.fecha_entrega)/30, signed int)=0, 1,
convert (DATEDIFF (detalle_tarjeta_farmacia.fecha_prox_entrega, detalle_tarjeta_farmacia.fecha_entrega)/30, signed int))),0) as prescripcion
from
    detalle_tarjeta_farmacia
where
    detalle_tarjeta_farmacia.cantidad_entregada>0
    and detalle_tarjeta_farmacia.fecha_entrega=$P{fecha}
    and detalle_tarjeta_farmacia.entregado=1
    and detalle_tarjeta_farmacia.procedencia= 'Morbilidad') prescripsiones_Morb
```

Figura 34: Prescripciones Morbilidad.

Para la siguiente sección, recetas y prescripciones despachadas dentro de 24 horas (Figura 35) se utilizan sub-consultas similares a la sección anterior pero se pregunta, si en el mismo día existió una atención ya sea por control SAPU o por una hora de atención normal (una reserva de hora). La sub-consulta que obtiene las prescripciones de los pacientes de morbilidad, despachadas dentro de las 24 horas se ilustra en la Figura 36.

MES: 6
AÑO: 2011
DÍA

MONITOREO DIARIO DI

N° Rp despachadas antes de l
24 horas

| | RC | RM | PC | PM | RC | RM | PC | PM |
|---|---|---|---|---|---|---|---|---|
| 1 | 582 | 267 | 2052 | 574 | 132 | 226 | 433 | 504 |
| 2 | 601 | 250 | 2301 | 539 | 148 | 219 | 527 | 495 |
| 3 | 483 | 275 | 1789 | 690 | 144 | 221 | 554 | 587 |
| 4 | 30 | 119 | 103 | 275 | 17 | 113 | 69 | 262 |
| 5 | 0 | 105 | 0 | 329 | 0 | 100 | 0 | 312 |
| 6 | 659 | 328 | 2316 | 737 | 156 | 274 | 474 | 658 |
| 7 | 612 | 276 | 2313 | 602 | 134 | 236 | 482 | 542 |

Figura 35: Segunda Sección.

```
(select IFNULL(sum(if( convert (DATEDIFF (detalle_tarjeta_farmacia.fecha_prox_entrega, detalle_tarjeta_farmacia.fecha_entrega)/30, signed int)=0, 1,
convert (DATEDIFF (detalle_tarjeta_farmacia.fecha_prox_entrega, detalle_tarjeta_farmacia.fecha_entrega)/30, signed int))),0) as prescripcion_cronico
from
    detalle_tarjeta_farmacia inner join receta_farmacia on
    receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and detalle_tarjeta_farmacia.entregado=1
    and detalle_tarjeta_farmacia.fecha_entrega= $P{fecha}
    and EXISTS
        (select reservahora.id_profesional
        from reservahora
        where reservahora.fecha= detalle_tarjeta_farmacia.fecha_entrega
        and reservahora.id_profesional= receta_farmacia.run_profesional
        and reservahora.id_paciente= receta_farmacia.run_paciente)
        and detalle_tarjeta_farmacia.entregado=1
        and detalle_tarjeta_farmacia.procedencia='Morbilidad'

    or receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
    and detalle_tarjeta_farmacia.cantidad_entregada>0
    and detalle_tarjeta_farmacia.entregado=1
    and detalle_tarjeta_farmacia.fecha_entrega= $P{fecha}
    and EXISTS
        (select control_sapu.folio
        from control_sapu
        where CAST(control_sapu.fecha_ingreso AS date) = detalle_tarjeta_farmacia.fecha_entrega
        and control_sapu.id_paciente = receta_farmacia.run_paciente)
        and detalle_tarjeta_farmacia.entregado=1
        and detalle_tarjeta_farmacia.procedencia= 'Morbilidad') as prescripciones_morb,
```

Figura 36: Sub-consulta Prescripciones Morbilidad, Despachas dentro de 24 Horas.

Esta consulta (Figura 36) no solo repite predicados también realiza una operación innecesaria, esta operación resaltada en el cuadro rojo retornara un valor igual a cero. Esto es ya que si un paciente es de procedencia morbilidad o crónica sus datos se encuentran (o deben encontrarse) solo en la tabla reserva_hora. Así como los datos de los pacientes de procedencia SAPU deben encontrarse solo en la tabla control_sapu. En estos casos se puede aplicar la optimización semántica y eliminar las secciones que no aportan al resultado. Esta consulta también se puede modificar para obtener el total de morbilidad, envés de realizarla dos veces para las procedencias de SAPU y morbilidad, y sumarlas. La sub-consulta resultante que genera el total de prescripciones para la procedencia morbilidad se muestra en la Figura 37.

```
(select count(detalle_tarjeta_farmacia.folio_receta_farmacia) as prescripciones_morb
from
    detalle_tarjeta_farmacia inner join receta_farmacia on
    receta_farmacia.folio= detalle_tarjeta_farmacia.folio_receta_farmacia
where
    detalle_tarjeta_farmacia.cantidad_entregada>0
    and detalle_tarjeta_farmacia.entregado=1
    and detalle_tarjeta_farmacia.fecha_entrega= '2011-07-01'
    and (detalle_tarjeta_farmacia.procedencia='Morbilidad'
        or detalle_tarjeta_farmacia.procedencia= 'SAPU')
    and (EXISTS
        (select reservahora.id_profesional
        from reservahora
        where reservahora.fecha= detalle_tarjeta_farmacia.fecha_entrega
        and reservahora.id_profesional= receta_farmacia.run_profesional
        and reservahora.id_paciente= receta_farmacia.run_paciente)
        OR
        EXISTS
        (select control_sapu.folio
        from control_sapu
        where CAST(control_sapu.fecha_ingreso AS date) = detalle_tarjeta_farmacia.fecha_entrega
        and control_sapu.id_paciente = receta_farmacia.run_paciente))) as prescripciones_morb,
```

Figura 37: Sub-consulta Prescripciones Morbilidad Total.

Para la tercera sección, número de recetas despachadas después de 24 horas (Figura 38) se realizan todas las sub-consultas de la sección 1 y 2, luego se realiza una resta para obtener los resultados requeridos, por ejemplo recetas_cronicos – recetas_cronicos_despachadas_dentro_24. Por lo que los cambios anteriores se aplican de igual forma a esta consulta.

MES: 6
AÑO: 2011

MONITOREO DIARIO DE RECETAS EN ATENCIÓN PRIMA

| DÍA | N° Rp despachadas antes de 24 horas | | | | N° Rp despachadas después de 24 horas | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | RC | RM | PC | PM | RC | RM | PC | PM | RC | RM | PC | PM |
| 1 | 582 | 267 | 2052 | 574 | 132 | 226 | 433 | 504 | 450 | 41 | 1619 | 70 |
| 2 | 601 | 250 | 2301 | 539 | 148 | 219 | 527 | 495 | 453 | 31 | 1774 | 44 |
| 3 | 483 | 275 | 1789 | 690 | 144 | 221 | 554 | 587 | 339 | 54 | 1235 | 103 |

Figura 38: Tercera Sección.

Para la cuarta y última sección (Figura 39) se utilizan las 6 sub-consultas que generan todas las recetas de la primera sección y 2 sub-consultas de la segunda sección que calculan el total de recetas despachadas dentro de 24 horas de los pacientes de morbilidad y SAPU, y al aplicar una formula se calcula el porcentaje de recetas entregadas dentro de 24 horas. En este calculo no se consideran las recetas de los pacientes crónicos dentro de las 24 horas ya que como se menciono anteriormente

los pacientes crónicos están constantemente retirando remedios incluso si no tuvieron una atención ese día.

MES: 6
AÑO: 2011
DÍA

MONITOREO DIARIO DE RECETAS EN ATENCION PRIMARIA

| | N° Rp despachadas antes de 24 horas | | | | N° Rp despachadas después de 24 horas | | | | % de recetas para pacientes ambulatorios dispensadas completas el mismo día de su emisión | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | RC | RM | PC | PM | RC | RM | PC | PM | RC | RM | PC | PM | |
| 1 | 582 | 267 | 2052 | 574 | 132 | 226 | 433 | 504 | 450 | 41 | 1619 | 70 | 95,17 |
| 2 | 601 | 250 | 2301 | 539 | 148 | 219 | 527 | 495 | 453 | 31 | 1774 | 44 | 96,36 |
| 3 | 483 | 275 | 1789 | 690 | 144 | 221 | 554 | 587 | 339 | 54 | 1235 | 103 | 92,88 |
| 4 | 30 | 119 | 103 | 275 | 17 | 113 | 69 | 262 | 13 | 6 | 34 | 13 | 95,97 |
| 5 | 0 | 105 | 0 | 329 | 0 | 100 | 0 | 312 | 0 | 5 | 0 | 17 | 95,24 |

Figura 39: Cuarta Sección Monitoreo Diario.

El siguiente paso es encontrar todos los predicados en común de estas consultas ya reformuladas, para aplicar la optimización basada en el MQP. A continuación se presenta la tabla número 5 con los predicados comunes. Para calcular el número total de sub-consultas ejecutadas se considero un mes de 30 días. Considere que ahora solo se ejecutan 930 sub-consultas y no 1.140 sub-consultas.

| Predicado | Tipo | Tipo consulta | Fc | Fs |
|---|---|---|---|---|
| cast(receta_farmacia.fecha as date)= ${fecha} | Igualdad | recetas | 0,0247(*30) | 0,0065(*30) |
| reservahora.fecha= ${fecha} | Igualdad | Dentro 24 moribi y cron. | 0,0096(*30) | 0,0012(*30) |
| detalle_tarjeta_farmacia.fecha_entrega=${fecha} | Igualdad | prescripciones | 0,0086(*30) | 0,00803(*30) |
| CAST(control_sapu.fecha_ingreso AS date)= ${fecha} | Igualdad | dentro 24 SAPU | 0,0053(*30) | 0,00046(*30) |
| detalle_tarjeta_farmacia.entregado=1 | Igualdad | todas | 1 | 1 |
| detalle_tarjeta_farmacia.cantidad_entregada>0 | Igualdad | todas | 1 | 0,9943495 |
| detalle_tarjeta_farmacia inner join receta_farmacia | Igualdad join | recetas | 0,87096774 | |
| detalle_tarjeta_farmacia.procedencia='Morbilidad' | Igualdad | morbilidad | 0,41935484 | 0,25465568 |
| detalle_tarjeta_farmacia.procedencia='SAPU' | Igualdad | SAPU | 0,41935484 | 0,14412611 |
| detalle_tarjeta_farmacia.procedencia='Crónico' | Igualdad | crónico | 0,38709677 | 0,60121821 |

| Predicado | Tipo | Tipo consulta | Fc | Fs |
|---|---|---|---|---|
| (cast(receta_farmacia.fecha as time)>='17:00:00' and dayname(cast(receta_farmacia.fecha as date))not in ('Friday', 'Saturday', 'Sunday')) or (cast(receta_farmacia.fecha as time)>='16:00:00' and dayname(cast(receta_farmacia.fecha as date))='Friday') or (cast(receta_farmacia.fecha as time)>='9:00:00' and (dayname(cast(receta_farmacia.fecha as date))='Saturday' or dayname(cast(receta_farmacia.fecha as date))='Sunday') | Igualdad | recetas | 0,38709677 | 0,22391064 |
| (cast(receta_farmacia.fecha as time)<'17:00:00' and dayname(cast(receta_farmacia.fecha as date))not in ('Friday', 'Saturday', 'Sunday')) or (cast(receta_farmacia.fecha as time)<'16:00:00' and dayname(cast(receta_farmacia.fecha as date))='Friday') or (cast(receta_farmacia.fecha as time)<'9:00:00' and (dayname(cast(receta_farmacia.fecha as date))='Saturday' or dayname(cast(receta_farmacia.fecha as date))='Sunday') | Igualdad | recetas | 0,38709677 | 0,77608936 |
| reservahora.id_paciente= receta_farmacia.run_ | Igualdad join | dentro 24 exists | 0,29032258 | |
| control_sapu.id_paciente = receta_farmacia.run_paciente | Igualdad join | dentro 24 exists | 0,16129032 | |

Tabla 6: Predicados en Común Monitoreo Diario.

De igual forma que en el reporte de estadística farmacia en la Figura 40 se muestra el multi-grafo inicial, es importante recordar al momento de realizar las acciones a realizar en el multi-grafo las heurísticas previamente difinidas.

Analizando la Tabla número 6 vemos que las primeras acciones a realizar son para los predicados de selección de “detalle_tarjeta_farmacia.entregado=1” “detalle_tarjeta_farmacia.cantidad_entregada>0” pues poseen Fc=1 (heurística número 1). Además crearemos un super-conjunto para las fechas considerando la heurística número 4, ya que sabemos que todas pertenecen al mismo mes. Al momento de realizar esta acción aprovecharemos de seleccionar solo los atributos necesarios de cada tabla, por ejemplo de control_SAPU solo necesitamos el folio, fecha_ingreso e id_paciente. El grafo resultante se muestra en la Figura 41.

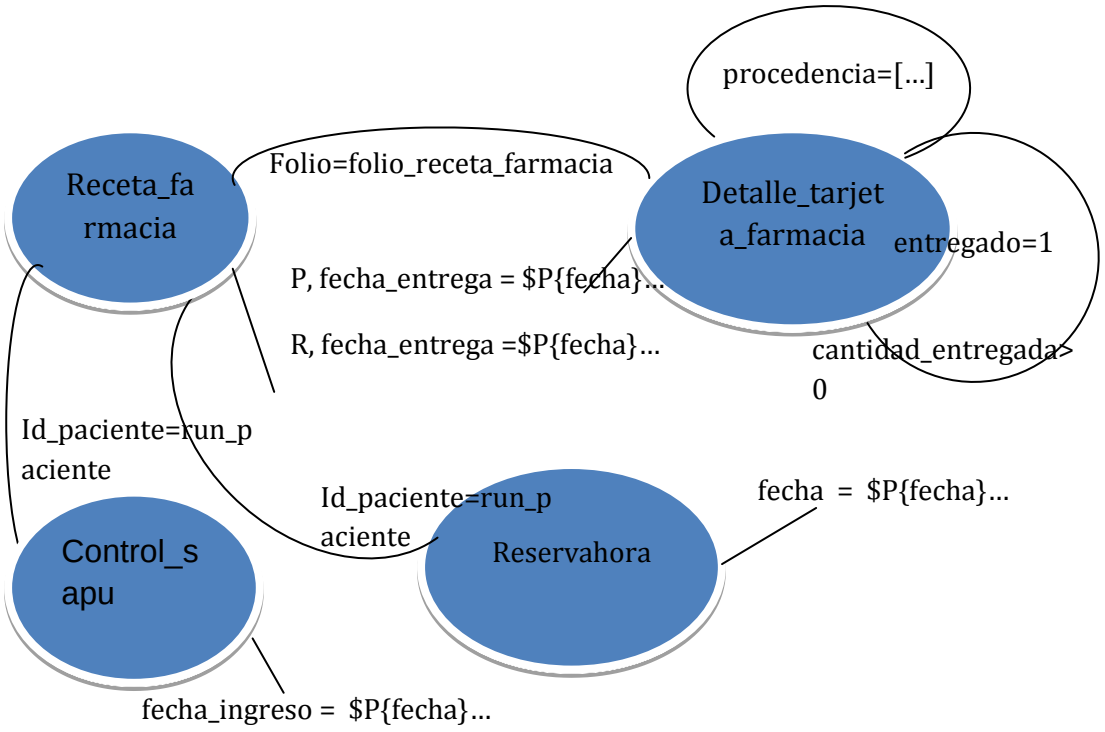



Figura 40: Multi-grafo Inicial Monitoreo Diario.

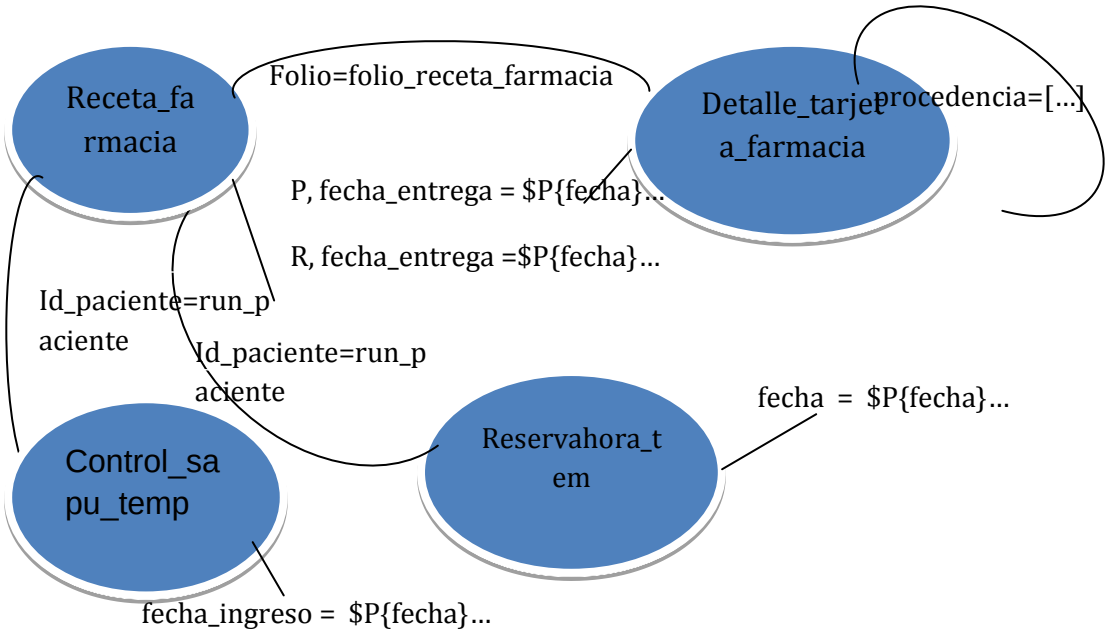



Figura 41: Multi-grafo Primera Iteración

Las siguientes acciones a aplicar sobre el multi-grafo serian para los predicados de procedencia. Pero en este caso, por conveniencia (tipos de consultas) se crearan dos grupos los de procedencia crónica y los de procedencia SAPU o morbilidad. Los

arcos correspondientes sobre detalle_tarjeta_farmacia pasarían a apuntar estos dos nuevos nodos. Esta acción se ve reflejada en la Figura 42.

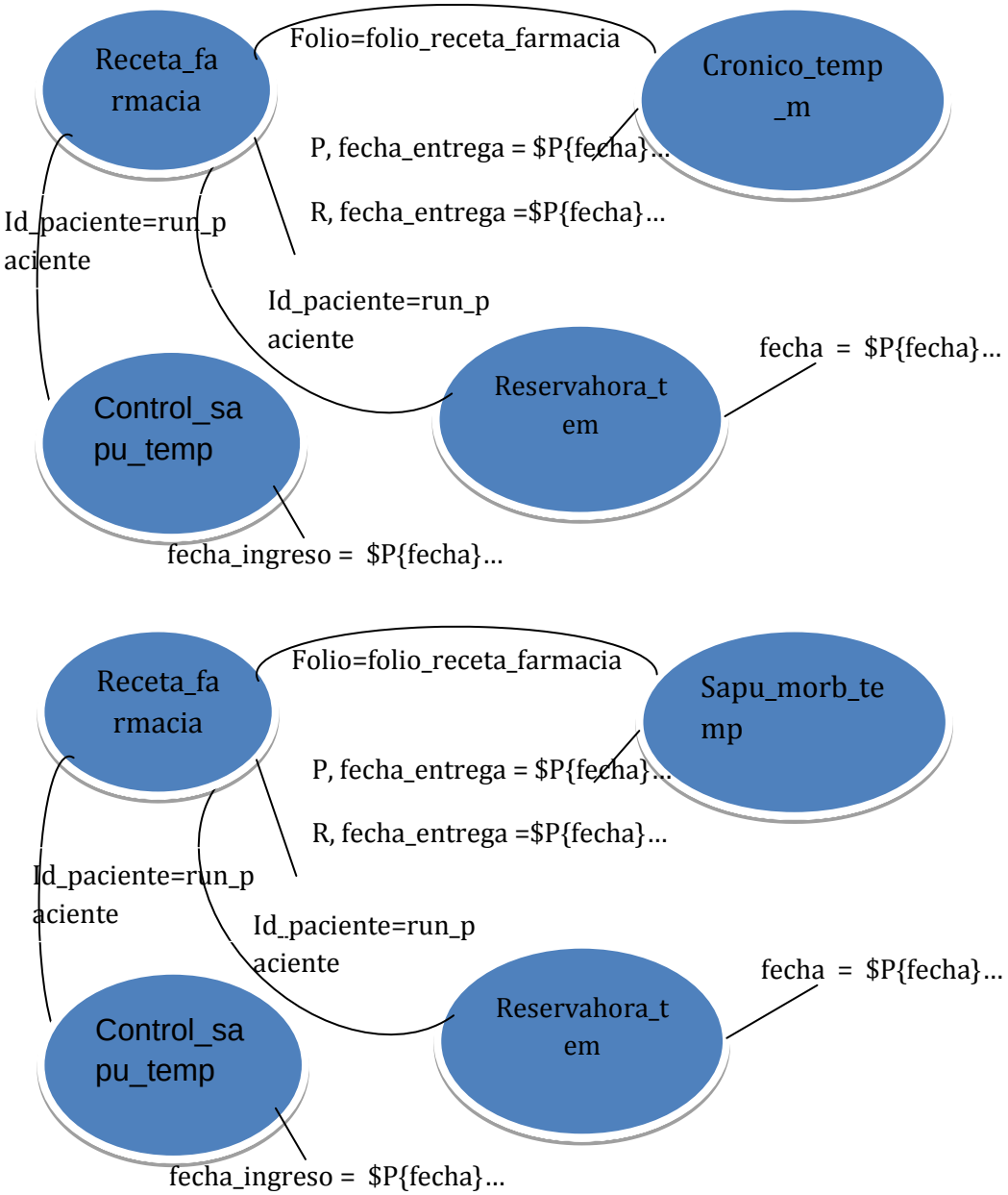



Figura 42: Multi-grafos Monitoreo Diario Segunda Iteración.

En este si aplicamos más predicados de selección significaría crear mas de 4 tablas, ya que poseemos 4 tablas temporales (sapu_mor_temp, cronico_temp, control_sapu_temp y reservahora_temp). Sin embargo podemos aplicar las acciones de join.

Respecto a los arcos de control_sapu_temp y reserva_hora_temp, estos no se efectuaran ya que basta con saber si existe o no la tupla correspondiente, por lo que

efectuar un join no es necesario y se utilizará la sentencia "Exists" cuando corresponda en cada consulta. El arco de join de receta_farmacia, con sapu_morb_temp y cronico_temp_m es ejecutado ya que afecta a un gran numero de consultas y se necesita más de una tributo de esta. Esta acción se ve reflejada en la Figura 43.

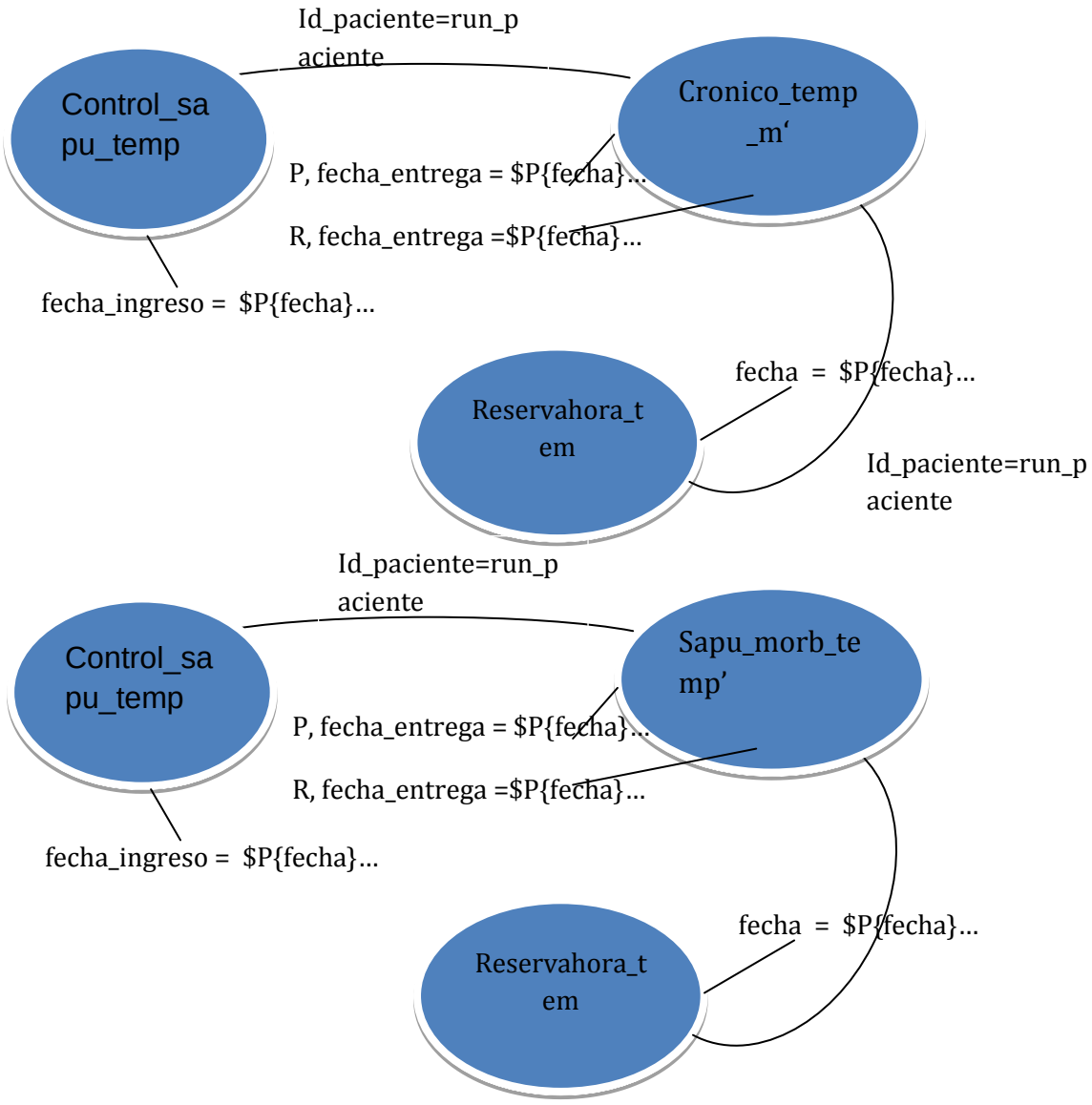



Figura 43: Multi-grafo Monitoreo Diario tercera iteración.

El siguiente paso es modificar las consultas y sub-consultas existentes, que trabajaran sobre estos resultados temporales, por ejemplo la sub-consulta que genera las recetas despachadas de morbilidad, dentro de 24 al ser modificada queda como se ilustra en la Figura 44.

```
(select count(DISTINCT(folio)) as receta_morbilidad
from
    `temp_sapu_morbi`
where
    `temp_sapu_morbi`.fecha_entrega=$P{fecha}
    and (EXISTS
        (select temp_reservahora.id_profesional
        from temp_reservahora
        where temp_reservahora.fecha= `temp_sapu_morbi`.fecha_entrega
        and temp_reservahora.id_profesional= `temp_sapu_morbi`.run_profesional
        and temp_reservahora.id_paciente= `temp_sapu_morbi`.run_paciente)
        OR
        EXISTS
        (select temp_control_sapu.folio
        from temp_control_sapu
        where CAST(temp_control_sapu.fecha_ingreso AS date) = `temp_sapu_morbi`.fecha_entrega
        and temp_control_sapu.id_paciente = `temp_sapu_morbi`.run_paciente))) as receta_morbilidad
```

Figura 44: Recetas Morbilidad Despachadas Dentro de 24 Horas.

## 6.3 Implementación de la solución

Para lograr implementar esta optimización, no se necesito modificar la estructura de los reportes, solo las consultas que los generaban. También se debió modificar la clase java que hacia la llamada a los reportes agregando dos funciones, una para insertar los datos en las tablas requeridas (según el reporte y las fechas entregadas por el usuario), antes de llamar al reporte y otra para eliminar los datos de estas tablas después de ser obtenido el reporte. Para evitar conflictos entre usuarios al momento de insertar e eliminar en estas tablas, se decidió utilizar el comando LOCK TABLES y bloquear las tablas para escrituras, esto limita su uso ya que solo un usuario a la vez podrá solicitar y hacer uso de las tablas.

La interfaz no se ve alterada, por lo que no ahí ningún cambio para el usuario final.

### 6.3.1 Script´s de las Tablas Creadas

A continuación se presentan los script´s de cada tabla, con sus respectivas funciones para insertar los datos en ellas. Primero se presentaran para el reporte de estadística farmacia y luego para el reporte de monitoreo diario.

#### *6.3.1.1 Reporte: Estadística Farmacia*

Para este reporte se crearon tres tablas para mantener los resultados, temporales las cuales se identifican como: temp_cronico, temp_morbilidad y temp_sapu

La primera tabla se presenta en la Figura 44 y su función para la inserción de datos en la Figura 45.

```
CREATE TABLE `temp_cronico` (
  `folio` varchar(20) CHARACTER SET latin1 NOT NULL DEFAULT '',
  `fecha_entrega` date NOT NULL DEFAULT '0000-00-00',
  `fecha_prox_entrega` date DEFAULT NULL,
  `hora` time DEFAULT NULL,
  `cantidad_entregada` double DEFAULT NULL,
  `fecha` datetime DEFAULT NULL,
  `run_paciente` varchar(20) CHARACTER SET latin1 DEFAULT NULL,
  `id_clasificacion` int(2) DEFAULT NULL,
  `numFamilia` varchar(9) CHARACTER SET latin1 DEFAULT NULL,
  `codigo_producto` int(20) NOT NULL DEFAULT '0',
  KEY `temp_sapu_idx1` (`folio`,`fecha`,`codigo_producto`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

Figura 44: Script de la Tabla temp_cronico

```
INSERT INTO `temp_cronico`()(select
        folio, fecha_entrega, fecha_prox_entrega, hora, cantidad_entregada,
        fecha, run_paciente, id_clasificacion, numFamilia, codigo_producto
    from
        (select folio_receta_farmacia,fecha_entrega, fecha_prox_entrega,
        hora, cantidad_entregada, codigo_producto
        from detalle_tarjeta_farmacia
        where detalle_tarjeta_farmacia.entregado = 1
        and `detalle_tarjeta_farmacia`.procedencia='Crónico') as detalle_1
        inner join `receta_farmacia` on folio_receta_farmacia=folio
        inner join paciente on paciente.run = receta_farmacia.run_paciente
    where
        (fecha_entrega between $P{fecha1} and $P{fecha2}
        or cast(fecha as date) between $P{fecha1} and $P{fecha2}))
```

Figura 45: Función Insert para temp_cronico.

La segunda tabla se presenta en la Figura 46 y su función para la inserción de datos en la Figura 47.

```
CREATE TABLE `temp_morbilidad` (
  `folio` varchar(20) CHARACTER SET latin1 NOT NULL DEFAULT '',
  `fecha_entrega` date NOT NULL DEFAULT '0000-00-00',
  `fecha_prox_entrega` date DEFAULT NULL,
  `hora` time DEFAULT NULL,
  `cantidad_entregada` double DEFAULT NULL,
  `fecha` datetime DEFAULT NULL,
  `run_paciente` varchar(20) CHARACTER SET latin1 DEFAULT NULL,
  `id_clasificacion` int(2) DEFAULT NULL,
  `numFamilia` varchar(9) CHARACTER SET latin1 DEFAULT NULL,
  `codigo_producto` int(20) NOT NULL DEFAULT '0',
  KEY `temp_sapu_idx1` (`folio`,`fecha`,`codigo_producto`),
  KEY `codigo_producto` (`codigo_producto`),
  CONSTRAINT `temp_morbilidad_fk1` FOREIGN KEY (`codigo_producto`) REFERENCES `producto_generico` (`Id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

Figura 46: Script de la Tabla temp_morbilidad.

```
INSERT INTO `temp_morbilidad`()(select
        folio, fecha_entrega, fecha_prox_entrega, hora, cantidad_entregada,
        fecha, run_paciente, id_clasificacion, numFamilia, codigo_producto
    from
        (select folio_receta_farmacia,fecha_entrega, fecha_prox_entrega,
        hora, cantidad_entregada, codigo_producto
        from detalle_tarjeta_farmacia
        where detalle_tarjeta_farmacia.entregado = 1
        and `detalle_tarjeta_farmacia`.procedencia='Morbilidad') as detalle_1
        inner join `receta_farmacia` on folio_receta_farmacia=folio
        inner join paciente on paciente.run = receta_farmacia.run_paciente
    where
        (fecha_entrega between $P{fecha1} and $P{fecha2}
        or cast(receta_farmacia.fecha as date) between $P{fecha1} and $P{fecha2}))
```

Figura 47: Función Insert para temp_morbilidad.

La tercera tabla se presenta en la Figura 48 y su función para la inserción de datos en la Figura 49.

```
CREATE TABLE `temp_sapu` (
  `folio` varchar(20) CHARACTER SET latin1 NOT NULL DEFAULT '',
  `fecha_entrega` date NOT NULL DEFAULT '0000-00-00',
  `fecha_prox_entrega` date DEFAULT NULL,
  `hora` time DEFAULT NULL,
  `cantidad_entregada` double DEFAULT NULL,
  `fecha` datetime DEFAULT NULL,
  `run_paciente` varchar(20) CHARACTER SET latin1 DEFAULT NULL,
  `id_clasificacion` int(2) DEFAULT NULL,
  `numFamilia` varchar(9) CHARACTER SET latin1 DEFAULT NULL,
  `codigo_producto` int(20) NOT NULL DEFAULT '0',
  KEY `temp_sapu_idx1` (`folio`,`fecha`,`codigo_producto`),
  KEY `codigo_producto` (`codigo_producto`),
  CONSTRAINT `temp_sapu_fk1` FOREIGN KEY (`codigo_producto`) REFERENCES `producto_generico` (`Id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

Figura 48: Script de la Tabla temp_sapu

```
INSERT INTO `temp_sapu`()(select
        folio, fecha_entrega, fecha_prox_entrega, hora, cantidad_entregada,
        fecha, run_paciente, id_clasificacion, numFamilia, codigo_producto
    from
        (select folio_receta_farmacia,fecha_entrega, fecha_prox_entrega,
        hora, cantidad_entregada, codigo_producto
        from detalle_tarjeta_farmacia
        where detalle_tarjeta_farmacia.entregado = 1
        and `detalle_tarjeta_farmacia`.procedencia='SAPU') as detalle_1
        inner join `receta_farmacia` on folio_receta_farmacia=folio
        inner join paciente on paciente.run = receta_farmacia.run_paciente
    where
        (fecha_entrega between $P{fecha1} and $P{fecha2}
        or cast(receta_farmacia.fecha as date) between $P{fecha1} and $P{fecha2}))
```

Figura 49: Función Insert para temp_sapu.

#### *6.3.1.2 Reporte Monitoreo Diario*

Para este reporte se crearon 4 tablas que mantendrán los resultados temporales, las cuales las identificamos como: temp_cronico_monitoreo, temp_sapu_morbi, temp_control_sapu.

La primera tabla se presenta en la Figura 50 y su función para la inserción de datos en la Figura 51.

```
CREATE TABLE `temp_cronico_monitoreo` (
  `folio` varchar(20) CHARACTER SET latin1 NOT NULL DEFAULT '',
  `fecha_entrega` date NOT NULL DEFAULT '0000-00-00',
  `fecha_prox_entrega` date DEFAULT NULL,
  `hora` time DEFAULT NULL,
  `cantidad_entregada` double DEFAULT NULL,
  `fecha` datetime DEFAULT NULL,
  `run_paciente` varchar(20) DEFAULT NULL,
  `run_profesional` varchar(11) DEFAULT NULL,
  KEY `temp_sapu_idx1_new` (`folio`,`fecha`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 AVG_ROW_LENGTH=121;
```

Figura 50: Script de la Tabla temp_cronico_monitoreo.

```
INSERT INTO `temp_cronico_monitoreo`(
   SELECT folio, fecha_entrega, fecha_prox_entrega, hora,cantidad_entregada, fecha, run_paciente, run_profesional
   FROM `receta_farmacia`inner join `detalle_tarjeta_farmacia` on folio= folio_receta_farmacia
   WHERE (detalle_tarjeta_farmacia.fecha_entrega between $P{fecha1} and $P{fecha2}
         or cast(receta_farmacia.fecha as date) between $P{fecha1} and $P{fecha2})
         and detalle_tarjeta_farmacia.entregado = 1
         and `detalle_tarjeta_farmacia`.`cantidad_entregada`>0
         and `detalle_tarjeta_farmacia`.`procedencia`= 'Crónico')
```

Figura 51: Función Insert para temp_cronico_monitoreo.

La segunda tabla se presenta en la Figura 52 y su función para la inserción de datos en la Figura 53.

```
CREATE TABLE `temp_sapu_morbi` (
  `folio` varchar(20) CHARACTER SET latin1 NOT NULL DEFAULT '',
  `fecha_entrega` date NOT NULL DEFAULT '0000-00-00',
  `fecha_prox_entrega` date DEFAULT NULL,
  `hora` time DEFAULT NULL,
  `cantidad_entregada` double DEFAULT NULL,
  `fecha` datetime DEFAULT NULL,
  `run_paciente` varchar(20) DEFAULT NULL,
  `run_profesional` varchar(11) DEFAULT NULL,
  `procedencia` varchar(10) DEFAULT NULL,
  KEY `temp_sapu_idx1_new_new` (`folio`,`fecha`),
  KEY `temp_sapu_morbi_idx1` (`run_paciente`),
  KEY `temp_sapu_morbi_idx2` (`run_profesional`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 AVG_ROW_LENGTH=121;
```

Figura 52: Script de la Tabla temp_sapu_morbi.

```
INSERT INTO `temp_sapu_morbi`(
   SELECT folio, fecha_entrega, fecha_prox_entrega, hora,cantidad_entregada, fecha, run_paciente, run_profesional,
   `detalle_tarjeta_farmacia`.`procedencia`
   FROM `receta_farmacia` inner join `detalle_tarjeta_farmacia` on folio= folio_receta_farmacia
   WHERE (detalle_tarjeta_farmacia.fecha_entrega between $P{fecha1} and $P{fecha2}
         or cast(receta_farmacia.fecha as date) between $P{fecha1} and $P{fecha2})
         and detalle_tarjeta_farmacia.entregado = 1
         and `detalle_tarjeta_farmacia`.`cantidad_entregada`>0
         and (`detalle_tarjeta_farmacia`.`procedencia`= 'SAPU'
         or `detalle_tarjeta_farmacia`.`procedencia`= 'Morbilidad'))
```

Figura 53: Función Insert para temp_sapu_morbi.

La tercera tabla se presenta en la Figura 54 y su función para la inserción de datos en la Figura 55.

```
CREATE TABLE `temp_control_sapu` (
  `folio` int(11) NOT NULL,
  `id_paciente` varchar(11) DEFAULT NULL,
  `id_medico` varchar(11) DEFAULT NULL,
  `fecha_ingreso` datetime DEFAULT NULL,
  PRIMARY KEY (`folio`),
  KEY `temp_control_sapu_idx1` (`id_paciente`),
  KEY `temp_control_sapu_idx2` (`id_medico`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 PACK_KEYS=0;
```

Figura 54: Script de la Tabla temp_sapu_morbi.

```
INSERT INTO `temp_control_sapu`(
    SELECT folio, id_paciente, id_medico, fecha_ingreso
    FROM `control_sapu`
    WHERE cast(`control_sapu`.`fecha_ingreso` as DATE) between $P{fecha1} and $P{fecha2})
```

Figura D.55: Función Insert para temp_control_sapu.

La cuarta tabla se presenta en la Figura 56 y su función para la inserción de datos en la Figura 57.

```
CREATE TABLE `temp_reservahora` (
  `folio` int(11) NOT NULL,
  `id_paciente` varchar(20) DEFAULT NULL,
  `fecha` date DEFAULT NULL,
  `id_profesional` varchar(60) DEFAULT NULL,
  PRIMARY KEY (`folio`),
  KEY `id_paciente` (`id_paciente`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 PACK_KEYS=0;
```

Figura 56: Script de la tabla temp_reservahora.

```
INSERT INTO `temp_reservahora`(
    SELECT folio, id_paciente, fecha, id_profesional
    FROM `reservahora`
    WHERE `reservahora`.`fecha` between $P{fecha1} and $P{fecha2})
```

Figura 57: Función Insert para temp_reservahora.

# Capítulo VII: Resultados de la Optimización

## 7.1 Introducción

Aunque no hemos seguido el algoritmo de MQP como es planteado por (Chen & Dunham, 1998) hemos logrado realizar cambios significativos. En este capítulo se presentan los resultados más significativos para cada reporte.

Además durante algunas pruebas se detecto que existía un error en el sistema de control de farmacia. La clase java al realizar la acción de llamar y conectar el reporte se ejecutaba dos veces. Lo que producía que el reporte se ejecutara dos veces. Este fallo ocurría en todos los reportes del sistema. Por lo que el tiempo real que tardaba el reporte de estadística farmacia y el reporte de monitoreo diario corresponden a 4:20 min. y a 7:45 min.

## 7.2 Reporte Estadística Farmacia

- El predicado "entregado=1" al igual que los predicados de procedencia = (crónico, morbilidad, SAPU) ahora solo se ejecutan 3 veces en total, anteriormente se ejecutaban en todas las consultas y sub-consultas (en total 888 veces).
- Los join´s de receta_farmacia y paciente anteriormente se ejecutaban 888 y 768 veces respectivamente, ahora solo se ejecutan 3 veces cada uno y sobre tablas reducidas tanto en columnas (por la proyección) como en tuplas (por la selección de procedencia).
- Las siguientes acciones de las distintas consultas y sub-consultas se ejecutaran sobre estas nuevas tablas que se encuentran reducidas en columnas y tuplas, como se menciono en el punto anterior, y reducidas por el super-conjunto creado para los datos de tipo fecha.

A continuación en la Tabla 7 se comparan algunos datos estadísticos de los reportes estadística farmacia original y final. Aunque se pude observar que se utiliza una mayor cantidad de tablas, pues se agregaron las tablas para mantener los resultados temporales, estas son utilizadas de una forma más eficiente. Lo que a producido una disminución significante en el tiempo promedio de obtención.

| | **Estadística farmacia** | **Esta. Farm. Nuevo** |
|---|---|---|
| Tiempo promedio de obtención | (8:30) min. | 42,5 seg. |
| Número de tablas involucradas | 4 | 7 |
| Consultas ejecutadas | 37 | 37 |
| Sub-consultas ejecutadas | 888 | 740 |
| Líneas de código promedio por consulta | 1112 | 281 |

Tabla 7: Datos Estadísticos del Reporte Estadística Farmacia.

A continuación en la Figura 58 se muestra un gráfico de barras que ilustra los tiempo (minutos : segundos) requerido para obtener el reporte de estadística farmacia, donde la **columna 1** representa el tiempo que tardaba originalmente este reporte en ser obtenido, la **columna 2** representa el tiempo real que tardaba este reporte (ya que se presentaba problemas no relacionados con las consultas SQL) y la **columna 3** que representa el tiempo que en que tarde en ser obtenido este reporte actualmente gracias a la optimización realizada en este estudio.

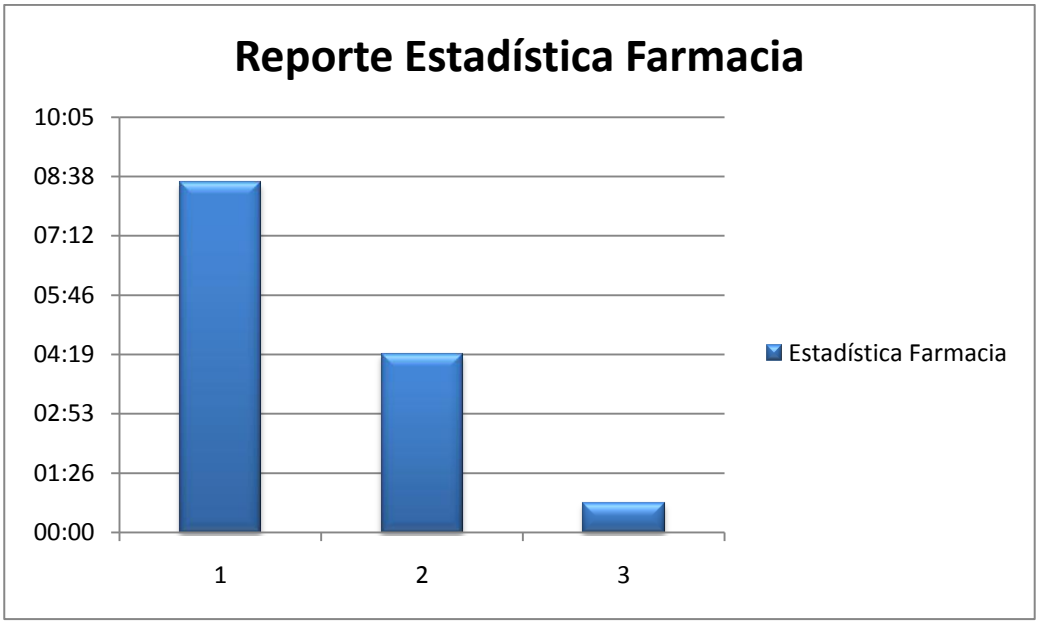



Figura 58: Tiempo de Obtención del Reporte de Estadística Farmacia.

## 7.3 Reporte Monitoreo Diario

Para el reporte de monitoreo diario se han logrado cambios significativos sin la necesidad de ejecutar todas las iteraciones para los multi-grafos generados para este reporte. Los cambios más significativos se deben a la creación de los super-conjuntos, según el mes solicitado, lo que produce una reducción significativa en los datos de cada tabla, como se menciono anteriormente esta base de datos se mantendrá creciendo durante el transcurso del tiempo y al reducir las tablas solo con los datos de un solo mes nos acercamos un tiempo de ejecución cercano a un valor constante, pues el resto de las consultas trabajaran con la misma cantidad de datos. En la Tabla número 8 se muestran algunos datos estadísticos de este reporte comparándolos con la situación inicial y final. Al igual que en caso anterior del reporte farmacia se han utilizado una mayor cantidad de tablas y se a logrado reducir el tiempo promedio de obtención gracias al pre-procesamiento de las consultas y las tablas temporales creadas.

| | Monitoreo diario | Monit. Diario Nuevo |
|---|---|---|
| Tiempo promedio de obtención | ( 15:30) min. | 1:31 min. |
| Número de tablas involucradas | 4 | 8 |
| Consultas ejecutadas | 120 | 120 |
| Sub-consultas ejecutadas | 1140 | 960 |
| Líneas de código por consulta (promedio) | 278 | 130 |

Tabla número 8: Datos Estadísticos del Reporte Moni toreo Diario.

A continuación en la Figura 59 se muestra un gráfico de barras que ilustra los tiempo (minutos : segundos) requerido para obtener el reporte de monitoreo diario, donde la **columna 1** representa el tiempo que tardaba originalmente este reporte en ser obtenido, la **columna 2** representa el tiempo real que tardaba este reporte (ya que se presentaba problemas no relacionados con las consultas SQL) y la **columna 3** que representa el tiempo que en que tarde en ser obtenido este reporte actualmente gracias a la optimización realizada en este estudio.

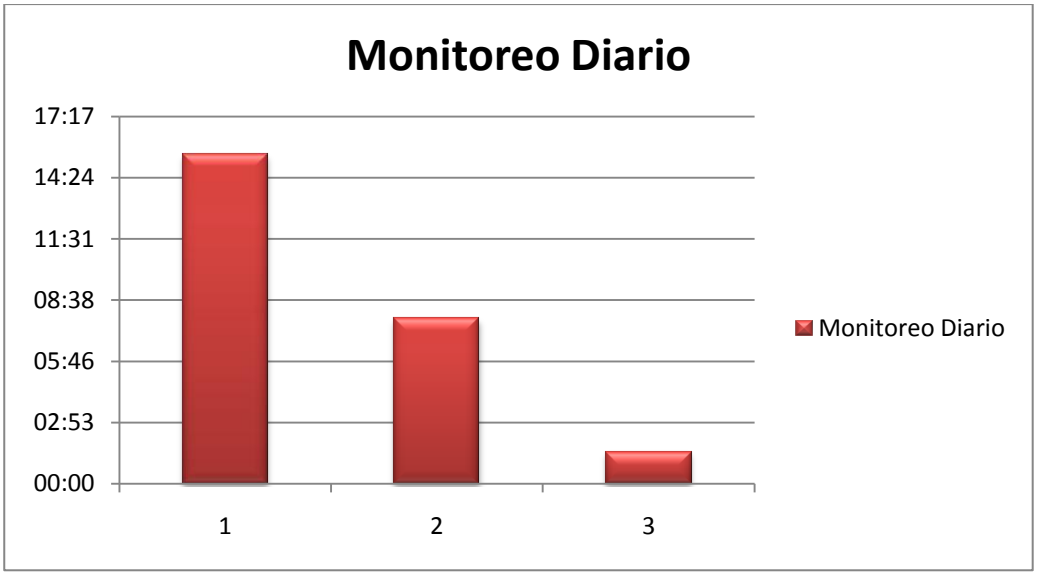



Figura 59: Tiempo de Obtención del Reporte de Monitoreo Diario.

Si bien las pruebas y resultados de este proyecto se basan en datos estadísticos, como el tiempo (que pueden variar debido cambios de hardware o uso del sistema), el verdadero resultado de esta optimización se ve reflejado en la reducción de los predicados ejecutados para la obtención de cada reporte y en la reducción de la cardinalidad (tamaño) de cada tabla con la que se trabaja, gracias al pre-procesamiento de las consultas y al almacenado de los resultados temporales. A continuación en la Tabla 9 se presenta la reducción de los predicados más significativos de cada consulta, indicando las veces que se ejecutaba anteriormente y

las veces que se ejecutan actualmente, para el reporte de estadística farmacia y en la Tabla 10 para el reporte de monitoreo diario.

| Predicado | Veces ejecutadas Ant. | Veces ejecutadas |
|---|---|---|
| entregado=1 | 740 | 3 |
| receta_farmacia inner join detalle_tarjeta_farmacia | 740 | 3 |
| detalle_tarjeta_farmacia inner join paciente on paciente | 560 | 3 |
| procedencia= 'Morbilidad' | 296 | 1 |
| procedencia= 'SAPU' | 296 | 1 |
| procedencia= 'Crónico' | 148 | 1 |
| fecha_entrega between $P{fecha1} and $P{fecha2} | 370 | Super-conjunto |
| cast(receta_farmacia.fecha as date) between $P{fecha} and $P{fecha} | 370 | |

Tabla 9: Reducción de Predicados del Reporte Estadística Farmacia.

| Predicado | Veces ejecutadas Ant. | Veces ejecutadas |
|---|---|---|
| entregado=1 | 930 | 2 |
| cantidad_entregada>0 | 930 | 2 |
| receta_farmacia inner join detalle_tarjeta_farmacia | 810 | 2 |
| procedencia= 'Morbilidad' | 390 | 5 |
| procedencia= 'SAPU' | 390 | 5 |
| procedencia= 'Crónico' | 360 | 1 |
| fecha_entrega = $P{fecha} | 240 | super_conjunto |
| cast(receta_farmacia.fecha as date) = $P{fecha} | 690 | |
| reservahora.fecha=$P{fecha}… | 279 | |

Tabla 10: Reducción de Predicados del Reporte Monitoreo Diario.

# Capítulo VIII: Conclusiones

## 8.1 Análisis de los Objetivos

Para nuestra hipótesis: "Al pre-procesar un conjunto de consultas de base de datos y aplicando una nueva estrategia de ejecución, para una aplicación en particular, se mejoran los tiempos de respuestas al detectar que consultas o las respuestas de algunas consultas (resultados intermedios) están contenidas dentro de otras consultas".

Podemos concluir que al pre-procesar las consultas y sus resultados intermedios (o predicados) se obtienen resultados favorables. Para un gran número de consultas en ejecución esto se puede lograr incluso sin un método eficiente para compartir los resultados intermedios.

Respecto al objetivo general y los objetivos específicos podemos concluir que se ha logrado con éxito la adaptación del servicio de "control de farmacia" del CESFAM Violeta Parra, elaborando una estrategia de ejecución diferente considerando una optimización para múltiples consultas y disminuyendo el tiempo total de estás. Además se logro con éxito el estudio de diferentes métodos de optimización y su implementación.

## 8.2 Principales Aportes

El principal aporte de este proyecto fue la demostración de la importancia de considerar la optimización al momento de realizar consultas a una base de datos, ya que siempre dejamos esta tarea al gestor de la base de datos. Ya sea una optimización individual por cada consulta o global para múltiples consultas, los beneficios pueden ser considerables.

Para el consultorio se lograron sus objetivos ya que se disminuyeron considerablemente los tiempos de obtención de cada reporte. Pasando de un tiempo de 08:30 min. a 00:43 min para el reporte de monitoreo estadística farmacia, lo que significa una mejora del 91% aproximadamente y de 15:30 min. a 01:31 min. el reporte de monitoreo diario lo que significa una mejora del 90% aproximadamente .

## 8.3 Trabajos Futuros

Para un mejor uso de este sistema se recomienda que se realicen cambios en es sistema de control de farmacia. Por ejemplo el uso de una receta electrónica haría que los datos fueran más precisos, el medicó podría crear la receta en el horario real y luego el paciente retiraría los productos en ventanilla solo con su rut, quedando registrados todos los datos requeridos (fecha, hora, etc.), Además con esto, se haría un mejor uso de la clave primaria ya que se podría generar una clave auto incremental y

no seria necesario el uso de las tablas de control_sapu o de reservahora para saber si se despacho el mismo día de ser emitida .

Otro cambio significativo que se podría realizar es respecto al control de los productos no despachados, ya que actualmente a los usuarios para despachar un producto pendiente se le da la opción de modificar la receta completa (eliminar productos anteriores, agregar productos nuevos, cambiar el estado de los productos entregados y por entregar), el realizar estas modificaciones pueden provocar errores al momento de calcular el stock o contar cada receta y prescripción despachada, ya que las fechas de creación y de retiro de recetas y prescripciones serán modificadas por estos cambios.

## Bibliografía

Abiteboul, S., Hull, R., & Vianu, V. (1995). *Foundations of Databases* (1° ed.). Boston, USA : Addison-Wesley Publishing Company, Inc.

Chen, F.-C. F., & Dunham, M. H. (1998). Common subexpression processing in multiple-query processing. *Transactions on*, 493–499.

Elmasri, R. A., & Navathe, S. (2002). *FUNDAMENTOS DE SISTEMAS DE BASES DE DATOS* (3° ed.). Madrid: Pearson educación.

Korth, H. F., & Silberschatz, A. (1993). *FUNDAMENTOS DE BASES DE DATOS* (2°. ed.). Madrid: McGraw-Hill.

Oracle and/or its affiliates. (20 de 12 de 2011). *MySQL 5.0 Reference Manual.* Obtenido de http://dev.mysql.com/doc/refman/5.0/en/

Tamer Ozsu, M., & Valduriez, P. (1999). *Principles of distributed database sistem* (2nd ed.). New Jersey: Prentice-Hall, Inc.

The Apache Software Foundation. Struts [en línea] < http://struts.apache.org/2.x/> [consulta: 04 de noviembre 2011].

Jasperforge. [en línea].<http://jasperforge.org/projects/ireport> [consulta: 04 de noviembre 2011].

MyEclipseide [en línea] <http://www.myeclipseide.com/> [consulta: 04 de noviembre 2011].

MySQL-Front [en línea]<http://www.mysqlfront.de/wp/> [consulta: 04 de noviembre].

MySQL. [en línea]<http://dev.mysql.com/> [consulta: 04 de noviembre de 2011].

Hp. Servidor HP ProLiant serie ML150 G6- Especificaciones [en línea]< http://h10010.www1.hp.com/wwpc/es/es/sm/WF06b/15351-15351-241434-3328424-3328424-3884323-3918837.html> [consulta: 28 de noviembre 2011].

BPMN. [en línea] <http://www.bpmn.org/> [consulta: 10 de enero de 2012].

## Anexo A: Tipos de Servidores

Un servidor es un computador o dispositivo en una red que administra los recursos de la red. Por ejemplo, un servidor de archivos es una computadora y un dispositivo de almacenamiento de dicado a almacenamiento de archivos. Cualquier usuario de la red puede almacenar archivos en el servidor. Un servidor de impresión es un equipo que maneja una o más impresoras. Un servidor de base de datos es un sistema informático que procesa las consultas de base de datos.

Los servidores dedicados significan que no realizan otras tareas además se sus de sus servidor. Sin embargo en los sistemas operativos multiprocesos una sola computadora puede ejecutar varios programas a la vez. Un servidor en este caso podría referirse al programa que es la gestión de los recursos en lugar de todo el equipo.

Desde una perspectiva de hardware, un servidor es simplemente un equipo de la red que está con figurado para compartir sus recursos o ejecutar las aplicaciones de los otros ordenadores de la red. Lo que hace que el término "servidor" sea doblemente confuso es que puede referirse tanto a hardware y software, es decir, que puede ser utilizado para descfribir un paquete de software que se ejecutan en un ordenador o el equipo en el que el software se está ejecutando.

Los tipos de servidores más comunes son los siguientes:

- **Servidores de Aplicaciones (*Application Servers*):** también llamado appserver. Este se encarga de todas las operaciones de las aplicaciones entre los usuarios y las aplicaciones de una organización. Los servidores de aplicaciones se utilizan generalmente para aplicaciones basadas en transacciones complejas. Para apoyar la necesidad de las aplicaciones un servidor de aplicaciones tiene que tener redundancia integrada, monitores de alta disponibilidad, alto rendimiento distribuidos los servicios de aplicaciones y soporte para acceso a bases de datos complejas.
- **Servidores FTP (*FTP Servers*):** Un servidor FTP es una aplicación de software que corre sobre el protocolo de intercambio de archivos: File Transfer Protocol (FTP), a través de Internet. FTP funciona de la misma manera como HTTP para la transferencia de páginas Web desde un servidor al navegador del usuario y SMTP para transferir correo electrónico a través de Internet en el que, como estas tecnologías, FTP utiliza el protocolo TCP de Internet/IP para permitir la transferencia de datos. FTP es más comúnmente utilizado para descargar un archivo desde un servidor a través de Internet o para subir un archivo a un servidor (por Ejemplo, subir un archivo de página Web en un servidor).

- **Servidores Web (*Web Servers*):** Básicamente, un servidor web sirve contenido estático a un navegador, carga un archivo y lo sirve a través de la red al navegador de un usuario. Este intercambio es mediado por el navegador y el servidor que hablan el uno con el otro mediante HTTP. Se pueden utilizar varias tecnologías en el servidor para aumentar su potencia más allá de su capacidad de entregar páginas HTML; éstas incluyen scripts CGI, seguridad SSL y páginas activas del servidor (ASP).
- **Servidores Groupware (*Groupware Servers*):** Un servidor groupware es un software diseñado para permitir colaborar a los usuarios, sin importar la localización, vía Internet o vía Intranet corporativo y trabajar juntos en una atmósfera virtual.
- **Servidores IRC (*IRC Servers*):** Otra opción para usuarios que buscan la discusión en tiempo real, Internet Relay Chat consiste en varias redes de servidores separadas que permiten que los usuarios conecten el uno al otro vía una red IRC.
- **Servidores Proxy (*Proxy Servers*):** Un servidor que se encuentra entre una aplicación cliente, como por ejemplo un navegador Web y un servidor real. Que intercepta todas las peticiones al servidor real para ver si se puede cumplir la misma solicitud. Si no es así, envía la solicitud al servidor real. Los servidores proxy pueden mejorar el rendimiento para grupos de usuarios. Esto se debe a que guarda los resultados de todas las solicitudes de una cierta cantidad de tiempo y también puede filtrar solicitudes para bloquear o no ciertos tipos específicos de solicitudes salientes o entrantes al servidor.
- **Servidores de Base de Datos (Database Server):** Un servidor de base de datos es una aplicación que se basa en el modelo de arquitectura cliente / servidor. La aplicación se divide en dos partes: un front-end que se ejecutan en una estación de trabajo (donde los usuarios recopilar y muestran la información de base de datos) y el back-end que se ejecutan en un servidor en las tareas como el análisis y almacenamiento de datos se llevan a cabo.

El tipo de sistema que se elija para un servidor depende principalmente de su aplicación dentro de su organización, la cantidad de datos que será responsable de almacenar y recuperar, el número de solicitudes de los usuarios que usted espera que se envíe al servidor, y el número de clientes que accede al servidor son todas las cosas que usted tendrá que considerar antes de elegir una arquitectura de servidor y un paquete de software.

# Anexo B: Algoritmos Básicos

(Elmasri & Navathe, 2002) Un SGBD debe contar con algoritmos para implementar los distintos tipos de operaciones relacionales que pueden aparecer en una estrategia de ejecución de consulta:

- Ordenación
- Selección
- Reunión
- Operaciones de agregación
- Operaciones de conjunto

## Ordenación

El algoritmo de ordenación es básico en el procesamiento de consultas. Se necesita cuando:

- Resultado ordenador (ORDER BY).
- Se desean eliminar duplicados (DISTINC).
- Operaciones de reunión.
- Y operaciones de conjuntos.

El algoritmo habitual para grandes ficheros de datos consiste en la estrategia ordenar-mezclar, que va ordenando pequeños ficheros y después los mezcla. El coste de estos algoritmos corresponde al coste de la ordenación (a*log(a)) y el coste de la mezcla (a).

## Selección

Los métodos de búsqueda para una condición simple se restringen por: una búsqueda lineal y el empleo de un solo índice. Para los métodos de búsqueda para una condición compleja, tenemos las siguientes opciones para las condiciones conjuntivas:

- Selección conjuntiva empleando un índice individual
- Selección conjuntiva con un índice compuesto.
- Selección conjuntiva por intersección de punteros a registros

Ejemplos de búsquedas, considere las siguientes relaciones:

Empleado(**nss**, nombre ,apellido, inic, fecha_naci, sexo, salario, nss_superv, nd)
Departamento(**numerod**, nombred, nss_jefe, fecha_inic_jefe)
Proyecto(**numerop**, nombrep, localizacionp, numd)
Trabaja_en(**nsse, np,** horas)

Para estas relaciones tenemos las siguientes posibilidades:

**Op1:** empleado.NSS = '1223567'
**Op2:** departamento.numerod > 5
**Op3:** empleado.nd=5
**Op4:** empleado.Nd =5 and empleado.salario>3000 and empleado.sexo ='M'
**Op5:** trabaja_en.nsse='123456' and trabaja_en.np=10

Se busca la mejor vía de acceso a los datos: si existe un índice se utiliza y si no se accede secuencialmente. Si la condición es compleja, se accede en primer lugar a la condición más restrictiva. Por ejemplo en Op4 se elegiría la más restrictiva

## Reunión

La reunión (join) es una de las operaciones más costosas, a continuación se listan las diferentes formas de abordarlos.

- Reunión de bucle anidado (secuencial).
- Reunión por bucle único (cuando existe alguna estructura de acceso (índice) para extraer los registros coincidentes).
- Reunión por ordenación-mezcla. Primero se ordenan las relaciones implicadas y después se mezclan.
- Reunión por dispersión: una de las relaciones operando se dispersa por el atributo de reunión y después se accede secuencialmente a la otra relación.
- Reunión con índice es una de las relaciones operando: la relación que no está indexada se accede secuencial-Mente y para cada tupla se accede a la otra relación a través del índice.
- Reunión con índice en las dos relaciones: se mezclan los dos índices para conseguir un conjunto de parejas (TID1, TID2) de tuplas coincidentes.
- Reunión por índice de reunión: es un índice especial para la realización de reuniones.

## Operaciones de agregación

Cuando los operadores de presentan funciones de agregación (MIN, MAX, COUNT, AVERAGE, SUMA), se pueden calcular mediante un acceso secuencial o mediante el índice adecuado. Ejemplo:

**select** max(salario)
**from** empleados;

Si existe índice en el campo salario se puede utilizar para encontrar el máximo.

**Operaciones de conjuntos.**

Estas operaciones tienen implementaciones costosas (unión, intersección, diferencia de conjuntos y producto cartesiano). La operación producto cartesiano debe evitarse por su elevado coste. Las otras tres operaciones en general se implementan mediante la técnica ordena-mezcla (nombrado anteriormente).

## Anexo C: Reporte Estadística Farmacia

# ESTADÍSTICA

Entr 01/07/20 y 31/07/20

**0-9 AÑOS**

| | SECTO R | SECTO P | SECTO R | SECTO P | SECTO R | SECTO P | SECTO R | SECTO P | SECTO R | SECTO P | SECTO R | SECTO P | SIN R | SIN P | TOTAL R | TOTAL P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Morbilidad Diurno | 188 | 354 | 220 | 436 | 30 | 59 | 88 | 166 | 113 | 214 | 28 | 49 | 45 | 108 | 712 | 1386 |
| Morbilidad Vespertino | 17 | 35 | 17 | 38 | 3 | 4 | 3 | 10 | 10 | 15 | 2 | 3 | 7 | 17 | 59 | 122 |
| **Total Morbilidad** | 205 | 389 | 237 | 474 | 33 | 63 | 91 | 176 | 123 | 229 | 30 | 52 | 52 | 125 | 771 | 1508 |
| Morbilidad Diurno No | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Morbilidad Vespertino No | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Total Morbilidad No** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| SAPU Diurno | 16 | 36 | 26 | 60 | 7 | 16 | 13 | 38 | 9 | 29 | 5 | 11 | 21 | 52 | 97 | 242 |
| SAPU Vespertino | 59 | 139 | 86 | 197 | 26 | 62 | 26 | 64 | 22 | 56 | 13 | 24 | 37 | 92 | 269 | 634 |
| **Total SAPU** | 75 | 175 | 112 | 257 | 33 | 78 | 39 | 102 | 31 | 85 | 18 | 35 | 58 | 144 | 366 | 876 |
| SAPU Diurno No Desp. | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| SAPU Vespertino | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Total SAPU No Desp.** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Crónico Diurno | 37 | 55 | 43 | 85 | 5 | 10 | 26 | 56 | 9 | 12 | 11 | 14 | 10 | 13 | 141 | 245 |
| Crónico Vespertino | 1 | 1 | 5 | 12 | 0 | 0 | 0 | 0 | 3 | 4 | 1 | 3 | 0 | 0 | 10 | 20 |
| **Total Crónico** | 38 | 56 | 48 | 97 | 5 | 10 | 26 | 56 | 12 | 16 | 12 | 17 | 10 | 13 | 151 | 265 |
| **Total Despachado** | 318 | 620 | 397 | 828 | 71 | 151 | 156 | 334 | 166 | 330 | 60 | 104 | 120 | 282 | 1288 | 2649 |
| **Total No Despachado** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**10-19 AÑOS**

| | SECTO R | SECTO P | SECTO R | SECTO P | SECTO R | SECTO P | SECTO R | SECTO P | SECTO R | SECTO P | SECTO R | SECTO P | SIN R | SIN P | TOTAL R | TOTAL P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Morbilidad Diurno | 65 | 122 | 113 | 216 | 5 | 8 | 33 | 67 | 45 | 76 | 14 | 22 | 27 | 48 | 302 | 559 |
| Morbilidad Vespertino | 22 | 38 | 29 | 59 | 5 | 8 | 23 | 33 | 18 | 36 | 4 | 4 | 10 | 19 | 111 | 197 |
| **Total Morbilidad** | 87 | 160 | 142 | 275 | 10 | 16 | 56 | 100 | 63 | 112 | 18 | 26 | 37 | 67 | 413 | 756 |
| Morbilidad Diurno No | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Morbilidad Vespertino No | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Total Morbilidad No** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| SAPU Diurno | 4 | 13 | 9 | 24 | 3 | 9 | 5 | 15 | 1 | 3 | 2 | 7 | 10 | 24 | 34 | 95 |
| SAPU Vespertino | 35 | 87 | 52 | 136 | 29 | 75 | 26 | 59 | 17 | 41 | 11 | 26 | 28 | 67 | 198 | 491 |
| **Total SAPU** | 39 | 100 | 61 | 160 | 32 | 84 | 31 | 74 | 18 | 44 | 13 | 33 | 38 | 91 | 232 | 586 |
| SAPU Diurno No Desp. | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| SAPU Vespertino | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Total SAPU No Desp.** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Crónico Diurno | 28 | 38 | 32 | 63 | 5 | 23 | 25 | 43 | 33 | 55 | 13 | 20 | 7 | 13 | 143 | 255 |
| Crónico Vespertino | 1 | 8 | 5 | 15 | 3 | 6 | 1 | 2 | 2 | 2 | 0 | 0 | 0 | 0 | 12 | 33 |
| **Total Crónico** | 29 | 46 | 37 | 78 | 8 | 29 | 26 | 45 | 35 | 57 | 13 | 20 | 7 | 13 | 155 | 288 |
| **Total Despachado** | 155 | 306 | 240 | 513 | 50 | 129 | 113 | 219 | 116 | 213 | 44 | 79 | 82 | 171 | 800 | 1630 |
| **Total No Despachado** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**20-64 AÑOS**

| | SECTO R | SECTO P | SECTO R | SECTO P | SECTO R | SECTO P | SECTO R | SECTO P | SECTO R | SECTO P | SECTO R | SECTO P | SIN R | SIN P | TOTAL R | TOTAL P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Morbilidad Diurno | 466 | 928 | 550 | 998 | 33 | 54 | 298 | 628 | 273 | 487 | 106 | 227 | 111 | 224 | 1837 | 3546 |
| Morbilidad Vespertino | 178 | 419 | 221 | 535 | 28 | 63 | 148 | 321 | 116 | 258 | 45 | 95 | 49 | 106 | 785 | 1797 |
| **Total Morbilidad** | 644 | 1347 | 771 | 1533 | 61 | 117 | 446 | 949 | 389 | 745 | 151 | 322 | 160 | 330 | 2622 | 5343 |
| Morbilidad Diurno No | 1 | 0 | 1 | 10 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 4 | 10 |
| Morbilidad Vespertino No | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Total Morbilidad No** | 1 | 0 | 1 | 10 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 4 | 10 |
| SAPU Diurno | 37 | 112 | 45 | 122 | 20 | 60 | 35 | 107 | 27 | 78 | 12 | 34 | 52 | 163 | 228 | 676 |
| SAPU Vespertino | 158 | 475 | 213 | 602 | 107 | 319 | 102 | 319 | 76 | 213 | 45 | 140 | 135 | 420 | 836 | 2488 |
| **Total SAPU** | 195 | 587 | 258 | 724 | 127 | 379 | 137 | 426 | 103 | 291 | 57 | 174 | 187 | 583 | 1064 | 3164 |
| SAPU Diurno No Desp. | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| SAPU Vespertino | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| **Total SAPU No Desp.** | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| Crónico Diurno | 1207 | 3788 | 1548 | 4860 | 89 | 131 | 1024 | 2946 | 822 | 2507 | 374 | 1283 | 393 | 1530 | 5457 | 17045 |
| Crónico Vespertino | 168 | 508 | 234 | 738 | 21 | 47 | 277 | 880 | 129 | 365 | 49 | 186 | 57 | 180 | 935 | 2904 |
| **Total Crónico** | 1375 | 4296 | 1782 | 5598 | 110 | 178 | 1301 | 3826 | 951 | 2872 | 423 | 1469 | 450 | 1710 | 6392 | 19949 |
| **Total Despachado** | 2214 | 6230 | 2811 | 7855 | 298 | 674 | 1884 | 5201 | 1443 | 3908 | 631 | 1965 | 797 | 2623 | 10078 | 28456 |
| **Total No Despachado** | 1 | 0 | 1 | 10 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 5 | 10 |

**65 AÑOS Y MÁS**

| | SECTO R | SECTO P | SECTO R | SECTO P | SECTO R | SECTO P | SECTO R | SECTO P | SECTO R | SECTO P | SECTO R | SECTO P | SIN R | SIN P | TOTAL R | TOTAL P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Morbilidad Diurno | 109 | 256 | 130 | 273 | 7 | 20 | 95 | 202 | 62 | 121 | 22 | 53 | 21 | 43 | 446 | 968 |
| Morbilidad Vespertino | 47 | 118 | 46 | 117 | 2 | 4 | 43 | 88 | 25 | 45 | 9 | 17 | 7 | 16 | 179 | 405 |
| **Total Morbilidad** | 156 | 374 | 176 | 390 | 9 | 24 | 138 | 290 | 87 | 166 | 31 | 70 | 28 | 59 | 625 | 1373 |
| Morbilidad Diurno No | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| Morbilidad Vespertino No | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Total Morbilidad No** | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| SAPU Diurno | 14 | 41 | 19 | 47 | 3 | 8 | 4 | 7 | 7 | 22 | 2 | 7 | 10 | 25 | 59 | 157 |
| SAPU Vespertino | 19 | 60 | 26 | 69 | 25 | 80 | 18 | 55 | 12 | 46 | 4 | 10 | 13 | 41 | 117 | 361 |
| **Total SAPU** | 33 | 101 | 45 | 116 | 28 | 88 | 22 | 62 | 19 | 68 | 6 | 17 | 23 | 66 | 176 | 518 |
| SAPU Diurno No Desp. | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| SAPU Vespertino | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Total SAPU No Desp.** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Crónico Diurno | 1136 | 4744 | 1300 | 5543 | 46 | 131 | 1247 | 5073 | 850 | 3362 | 246 | 1001 | 215 | 999 | 5040 | 20853 |
| Crónico Vespertino | 126 | 458 | 221 | 887 | 6 | 12 | 280 | 1304 | 122 | 522 | 44 | 151 | 35 | 165 | 834 | 3499 |
| **Total Crónico** | 1262 | 5202 | 1521 | 6430 | 52 | 143 | 1527 | 6377 | 972 | 3884 | 290 | 1152 | 250 | 1164 | 5874 | 24352 |
| **Total Despachado** | 1451 | 5677 | 1742 | 6936 | 89 | 255 | 1687 | 6729 | 1078 | 4118 | 327 | 1239 | 301 | 1289 | 6675 | 26243 |
| **Total No Despachado** | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

**RESUMEN**

| | 0-9 AÑOS | | 10-19 AÑOS | | 20-64 AÑOS | | 65 AÑOS Y MÁS | | TOTAL | |
|---|---|---|---|---|---|---|---|---|---|---|
| | R | P | R | P | R | P | R | P | R | P |
| Morbilidad Diurno | 712 | 1386 | 302 | 559 | 1837 | 3546 | 446 | 968 | 3297 | 6459 |
| Morbilidad Vespertino | 59 | 122 | 111 | 197 | 785 | 1797 | 179 | 405 | 1134 | 2521 |
| **Total Morbilidad** | 771 | 1508 | 413 | 756 | 2622 | 5343 | 625 | 1373 | 4431 | 8980 |
| Morbilidad Diurno No | 0 | 0 | 0 | 0 | 4 | 10 | 1 | 0 | 5 | 10 |
| Morbilidad Vespertino No | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Total Morbilidad No** | 0 | 0 | 0 | 0 | 4 | 10 | 1 | 0 | 5 | 10 |
| SAPU Diurno | 97 | 242 | 34 | 95 | 228 | 676 | 59 | 157 | 418 | 1170 |
| SAPU Vespertino | 269 | 634 | 198 | 491 | 836 | 2488 | 117 | 361 | 1420 | 3974 |
| **Total SAPU** | 366 | 876 | 232 | 586 | 1064 | 3164 | 176 | 518 | 1838 | 5144 |
| SAPU Diurno No Desp. | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| SAPU Vespertino | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| **Total SAPU No Desp.** | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| Crónico Diurno | 141 | 245 | 143 | 255 | 5457 | 17045 | 5040 | 20853 | 10781 | 38398 |
| Crónico Vespertino | 10 | 20 | 12 | 33 | 935 | 2904 | 834 | 3499 | 1791 | 6456 |
| **Total Crónico** | 151 | 265 | 155 | 288 | 6392 | 19949 | 5874 | 24352 | 12572 | 44854 |
| **Total Despachado** | 1288 | 2649 | 800 | 1630 | 10078 | 28456 | 6675 | 26243 | 18841 | 58978 |
| **Total No Despachado** | 0 | 0 | 0 | 0 | 5 | 10 | 1 | 0 | 6 | 10 |