



Instituto Tecnológico Autónomo de México

Maestría en Ciencias en Computación

Minería de Datos

Dr. Ángel Kuri

Otoño 2015

Presentación 1: *Bases de datos relacionales*

Fabiola Cerón Flores 36027

María Fernanda Mora Alba 103596

Fernanda Alcalá Durand 118716

Sergio Haro Pérez - 148076

Norma Verónica Trinidad Hernández 156870

Elisa Hernández Rodríguez 160051

Antecedentes

Fue propuesto en 1970 por Edgar Frank Codd (IBM) lo cual desencadenó el diseño de numerosos sistemas de gestión de bases de datos (SGBD o DBMS) basados en su modelo así como estudios teóricos que le dieron mayor fundamento y solidez.

La popularidad de los sistemas de gestión de datos relacionales y su éxito comercial se basó en el reconocimiento de los beneficios y sencillez en el manejo de datos del mundo corporativo estandarizando su uso en los 80's. Una vez consolidado el modelo, surge SEQUEL y SQL como lenguaje de consulta para bases de datos relacionales el cual se basa en el manejo del álgebra y cálculo relacional.

Actualmente es el modelo más utilizado en aplicaciones comerciales de procesamiento de datos.

Codd identifica dos principios en el manejo de información compartida:

1. *Independencia de los datos* tanto física como lógica, que implica que las aplicaciones que hagan uso de la base de datos deben mantenerse independientes de la forma en la que la base de datos se organiza. Es decir, la modificación de los elementos de la base de datos no debe modificar las aplicaciones ni su manipulación lógica.
2. *Consistencia de los datos*, la cual es asegurada a través de una serie de reglas para eliminar cualquier redundancia en el diseño de la base de datos utilizando técnicas matemáticas basadas en lógica de predicados. Lo anterior es el antecedente de la normalización de bases de datos (proceso de organización de atributos y tablas de una base de datos relacional para minimizar la redundancia de datos)

Modelo Relacional

El modelo de Codd consta de 3 partes:

1. Un modelo de datos
2. Los medios para expresar las operaciones en un lenguaje
3. Un conjunto de reglas de diseño que evitan los problemas de redundancia de datos.

Es un modelo para diseño y gestión de bases de datos que está basado en la lógica de predicados (o lógica de primer orden) y teoría de conjuntos. En él, toda la información se presenta en conjuntos de datos llamados tuplas agrupadas en relaciones.

El modelo relacional presenta datos almacenados en tablas (**relaciones**) representadas por un número variable de registros o filas (**tuplas**) y un número fijo de columnas (**atributos**), donde cada fila establece una relación entre un conjunto de valores y cada columna tiene un nombre único. Además describe una serie de operaciones lógicas que pueden ser utilizadas sobre los datos las cuales incluyen los medios para obtener subconjuntos de filas, columnas o datos de la combinación de diferentes tablas.

Representación Gráfica del Modelo Relacional

RELACIÓN

Atributo ₁	Atributo ₂	...	Atributo _n	
Valor _{1,1}	Valor _{1,2}	...	Valor _{1,n}	→ Tupla ₁
Valor _{2,1}	Valor _{2,2}	...	Valor _{2,n}	→ Tupla ₂
.
.
.
Valor _{m,1}	Valor _{m,2}	...	Valor _{m,n}	→ Tupla _m

De esta forma:

- Relación es una tabla
- Tupla es una fila de la tabla
- Atributo es una columna de la tabla
- Cardinalidad es el número de tuplas de la tabla (m) y es variable
- Grado es el número de atributos de la tabla (n) y es invariable
- Dominio es el conjunto válido de valores que puede tener un atributo y puede ser texto, números enteros, flotantes, fecha, hora, booleano, etc.. El valor nulo indica que el valor es desconocido o no existe y siempre es miembro del dominio.

El nombre de modelo relacional se refiere a la estrecha relación que existe entre el elemento básico de este modelo, la tabla, y el concepto matemático de relación ya que una tupla representa la relación entre el conjunto de valores que toma cada atributo y una tabla es un conjunto de tuplas. Matemáticamente una tupla es una secuencia de valores

Estructura de las Bases de Datos Relacionales

Sea r una relación y los conjuntos D_1, D_2, \dots, D_n , un conjunto de dominios cualesquiera.

Matemáticamente se define una relación r sobre los conjuntos D_1, D_2, \dots, D_n como un subconjunto del producto cartesiano de la lista de dominios.

$$D_1 \times D_2 \times \dots \times D_n$$

En el modelo relacional una relación (o tabla) contiene un subconjunto de todas las tuplas (filas) posibles, por lo que una tabla de n atributos será un subconjunto de $D_1 \times D_2 \times \dots \times D_n$.

Por lo anterior, el orden de las tuplas en una relación no es relevante ya que ordenadas o no, la relación es la misma ya que contiene el mismo conjunto de tuplas.

Una tupla t de una relación r con n atributos se denota como $t(v_1, v_2, \dots, v_n)$ donde $v_1 \in D_1, v_2 \in D_2, \dots, v_n \in D_n$

Se dice que $t \in r$ cuando la tupla t está en la relación r .

$\forall r$ los dominios de todos los atributos deben ser atómicos; es decir, que los elementos del dominio son indivisibles (ej: el atributo "No_Teléfono" debe contener un solo teléfono, no un conjunto de teléfonos).

Álgebra Relacional

El álgebra relacional define un conjunto de operaciones sobre relaciones que funcionan de forma similar a las operaciones algebraicas comunes con números (tales como suma, resta o multiplicación). Las operaciones del álgebra relacional requieren números como insumo y generan un nuevo número mientras que las operaciones del álgebra relacional utilizan relaciones para generar una nueva relación.

El álgebra relacional es de tipo procedimental ya que describe paso a paso cómo obtener una respuesta sobre una base de datos.

Es importante que el resultado de una operación de álgebra relacional sea una nueva relación puesto que los argumentos de las operaciones fundamentales siempre son relaciones y lo anterior nos permite hacer composiciones para formar **expresiones** de álgebra relacional (tales como una proyección de una selección previa). Se puede hacer uso de las operaciones fundamentales tanto con relaciones como con expresiones.

A continuación se mencionan las operaciones fundamentales del álgebra relacional:

- Selección (σ): devuelve las tuplas de una relación que satisfacen una cierta condición (predicado). Notación: $\sigma_{condicion}(r)$
- Proyección (Π): devuelve los atributos especificados en una lista de atributos de todas las tuplas de una relación removiendo las tuplas duplicadas. Notación: $\Pi_{atributo_1, \dots, atributo_n}(r)$
- Unión (\cup): genera la unión de tuplas de las dos relaciones utilizadas como argumento. Notación: $r_1 \cup r_2$
Nota: Las relaciones r_1 y r_2 deben tener el mismo grado (número de atributos, n) y los dominios de sus atributos ser iguales para todo atributo.
- Diferencia de conjuntos ($-$): devuelve las tuplas que están en r_1 pero no están en r_2 . Notación: $r_1 - r_2$
- Producto Cartesiano (\times): genera todos los pares posibles de tuplas de dos relaciones. Si r_1 tiene n tuplas y r_2 tiene m tuplas el resultado de esta operación tendrá nxm tuplas. Notación: $r_1 \times r_2$
- Renombramiento (ρ): genera el resultado de una expresión con un nombre x . Notación: $\rho_x(r)$

Cálculo Relacional de Tuplas

Es un lenguaje de consulta declarativo, no procedimental ya que no describe paso a paso cómo obtener una respuesta sobre una base de datos; es decir, sólo refiere la notación para definir la operación deseada. Está basado en la lógica de predicados o de primer orden.

Las consultas de cálculo relacional de tuplas emplean la siguiente notación:

De manera general $\{variables | condicion\}$ y de manera específica $\{t | F(t)\}$

Donde:

t representa una variable de tupla (una variable que representa a una tupla; en otras palabras, una tupla que representa al conjunto de todas las tuplas posibles de una relación) y,

$F(t)$ es una fórmula/ condición/ predicado

De esta manera, la expresión de consulta representa el conjunto de tuplas que cumplen cierta condición.

Se dice que una variable tupla t es libre cuando no está cuantificada por \forall ó \exists , en cuyo caso es una variable tupla ligada.

Las fórmulas o condiciones $F(t)$ pueden ser construidas como fórmulas atómicas:

- $t \in r$ donde t es una variable tupla y r es una relación (no se permite el uso del operador \notin)
- $s[x] \Theta t[y]$ donde s y t son variables tuplas, x es un atributo de s , y es un atributo de t y Θ es un operador de comparación como $<, \leq, =, \neq, >, \geq$
- $t[x] \Theta c$ donde t es una variable tupla, x es un atributo de t , Θ es un operador de comparación como $<, \leq, =, \neq, >, \geq$ y c es una constante

Se utiliza la notación de la lógica de predicados: $\exists, \forall, \in, \wedge, \vee, \neg, \Rightarrow$.

Las fórmulas deben seguir ciertas reglas:

- Un átomo es una fórmula
- Si $F(t)$ es una fórmula, también lo es $\neg F(t)$ y $(F(t))$
- Si $F(t)$ y $G(t)$ son fórmulas, también lo son $F(t) \wedge G(t)$, $F(t) \vee G(t)$ y $F(t) \Rightarrow G(t)$
- Si $F(t)$ es una fórmula que contiene una variable tupla t y r es una relación se tiene que $\exists t \in r (F(t))$ y $\forall t \in r (F(t))$ también son fórmulas.

Elementos (Relaciones, Dominios, Claves)

Dominio: Conjunto de valores permitidos para cada atributo.

Relaciones: Subconjuntos del producto cartesiano de la lista de dominios.

Claves: Una clave permite identificar un conjunto de atributos suficiente para distinguir las entidades entre sí. Las claves también ayudan a identificar unívocamente a las relaciones y así a distinguir las relaciones entre sí.

Correspondencia de cardinalidades:

Uno a uno. Una entidad en A se asocia con a lo sumo una entidad en B, y una entidad en B se asocia con a lo sumo una entidad en A

Uno a varios. Una entidad en A se asocia con cualquier número de entidades en B (ninguna o varias). Una entidad en B, sin embargo, se puede asociar con a lo sumo una entidad en A

Varios a uno. Una entidad en A se asocia con a lo sumo una entidad en B. Una entidad en B, sin embargo, se puede asociar con cualquier número de entidades (ninguna o varias) en A

Varios a varios. Una entidad en A se asocia con cualquier número de entidades (ninguna o varias) en B, y una entidad en B se asocia con cualquier número de entidades (ninguna o varias) en A

Lenguajes de Consulta

los lenguajes de consulta se usan para especificar las solicitudes de información

- Álgebra relacional. El álgebra relacional forma la base del lenguaje de consulta SQL ampliamente usado.
- Cálculo relacional de tuplas
- Cálculo relacional de dominios

El cálculo relacional de dominios y tuplas, son lenguajes declarativos de consulta basados en la Lógica Matemática.

Los lenguajes se pueden clasificar en:

- Procedimentales. el usuario instruye al sistema para que lleve a cabo una serie de operaciones en la base de datos para calcular el resultado deseado.
- No procedimentales. el usuario describe la información deseada sin dar un procedimiento concreto para obtener esa información

La mayor parte de los sistemas comerciales de bases de datos relacionales ofrecen un lenguaje de consulta que incluye elementos de los enfoques procedimental y no procedimental.

Lenguaje SQL

SQL usa una combinación de álgebra relacional y construcciones del cálculo relacional.

IBM desarrolló la versión original en su Laboratorio de Investigación de San José . IBM implementó el lenguaje, originalmente denominado Sequel, como parte del proyecto System R, a principios de 1970. El lenguaje Sequel ha evolucionado desde entonces y su nombre ha pasado a ser SQL (Structured Query Language, Lenguaje estructurado de consultas). SQL se ha establecido como el lenguaje estándar de bases de datos relacionales.

La estructura básica de SQL es como sigue:

```
select A1, A2,..., An  
from r1, r2,..., rm  
where P
```

- La cláusula **select** corresponde a la operación proyección del álgebra relacional. Se usa para listar los atributos deseados del resultado de una consulta.
 - La cláusula **from** corresponde a la operación producto cartesiano del álgebra relacional. Lista las relaciones que deben ser analizadas en la evaluación de la expresión.
 - La cláusula **where** corresponde al predicado selección del álgebra relacional. Es un predicado que engloba a los atributos de las relaciones que aparecen en la cláusula from.
- OTROS Lenguajes. QBE (Query By Example), DATALOG

Gestores de Bases de Datos

El sitio <http://db-engines.com/en/ranking/relational+dbms> muestra un rank de alrededor de 100 gestores de base de datos de 280 que tiene registrado el sitio.

Entre los primeros 10 lugares de SGBD relacionales (RDMS) por popularidad están:

1. Oracle
2. MySql
3. Microsoft SQL Server
4. Postgresql
5. DB2
6. Access
7. SQLite
8. Teradata
9. Informix
10. Maria DB

Las 12 Reglas de Codd

Codd publica en los 80's 12 reglas que debe cumplir todo sistema de gestión de bases de datos (SGBD) para ser considerado relacional. Es importante mencionar que ninguno de los SGBD más populares cumplen con todas las reglas y sin embargo son considerados relacionales. Sin embargo, estas reglas son útiles como visión general de qué son los SGBD relacionales.

0. *El sistema debe ser relacional* (tanto Base de Datos como Manejador de Bases de Datos). Es decir, debe utilizar el modelo relacional (exclusivamente) para manejar la base de datos.
1. *Regla de la Información*. Toda la información de la base de datos debe estar representada explícitamente como valor de una columna dentro de una fila que pertenece a una tabla.
2. *Regla del Acceso garantizado*. Todo dato es accesible conociendo cuál es su clave primaria, la tabla y la columna que contiene el dato.
3. *Tratamiento sistemático*. El SGBD debe permitir el tratamiento adecuado de valores nulos distinto de valores regulares.
4. *Catálogo en línea basado en el modelo relacional*. La estructura de la base de datos debe ser accesible usando un esquema relacional.
5. *Sub-lenguaje de datos completo*. Al menos debe de existir un lenguaje que permita el manejo completo de la base de datos. Este lenguaje, por lo tanto, debe permitir realizar cualquier operación.
6. *Actualización de vistas*. El SGBD debe encargarse de que las vistas muestren la última información.
7. *Inserciones, modificaciones y eliminaciones de alto nivel*. Cualquier operación de modificación debe actuar sobre conjuntos de tuplas, nunca deben actuar registro a registro.
8. *Independencia física*. Los datos deben de ser accesibles desde la lógica de la base de datos aún cuando se modifique el almacenamiento.

9. *Independencia lógica.* Los programas no deben verse afectados por cambios en las tablas.
10. *Independencia de integridad.* Las reglas de integridad deben almacenarse en la base de datos, no en los programas de aplicación.
11. *Independencia de la distribución.* La distribución de la base de datos debe ser invisible al usuario.
12. *No subversión.* Si el SGBD posee un lenguaje que permite el recorrido registro a registro, éste no puede utilizarse para incumplir las reglas relacionales.

Integridad y Seguridad

Las bases de datos dentro del objetivo de almacenar, organizar y recuperar información, deben cumplir con diferentes condiciones técnicas, como memoria, espacio en disco duro, sistema de archivos. Y esto será dependiente de las configuraciones específicas del RDMS que se use. Existe un mínimo necesario de propiedades para proveer una base de datos como parte de una aplicación y su funcionamiento sea viable; conocido como ACID (por sus siglas en inglés, Atomicity, Consistency, Isolation and Durability)

Atomicidad: Si una operación consiste en una serie de pasos, todos ellos ocurren o ninguno, esto ayuda por ejemplo en cortes de energía, fallas de hardware, o eventos catastróficos.

Consistencia: Significa que cualquier estado de la base de datos será internamente consistente con las reglas que limitan los datos. P.ej. evitar elementos duplicados por llave primaria, o conjunto de llaves.

Aislamiento: Esto significa que puedes modificar diferentes partes de la base de datos al mismo tiempo sin afectar otra. El aislamiento es una parte fundamental y cada RDMS cuenta con opciones para aislar información.

Durabilidad: Significa que las otras acciones persisten una vez que las transacciones son completadas.

Si se garantizan estos puntos, se evitará muchas frustraciones en el uso de los datos .

Por qué son importantes

Funciones de las BDR

Existen diversas maneras de organizar la información de una base de datos: orientada a objetos, jerárquico, red, relacional, entre otros. Las BDR nos permiten, pues, organizar la información mediante tablas que están relacionadas.

Ventajas

Las BDR ofrecen ventajas a los individuos involucrados en su construcción y en su uso:

Facilidad de Construcción: generalidad sin dejar de ser simples, ya que la representación en tablas permite presentar la información de forma compacta

Facilidad de uso: el tener la información organizada y ordenada en tablas con columnas y filas es intuitiva para el usuario permite acceder fácilmente a información.

Flexibilidad en las consultas: es posible realizar consultas diversas y personalizadas gracias a que la información se puede extraer y manipular de las tablas usando operaciones hasta obtener la consulta deseada.

Precisión y alcance en las consultas: el uso de *álgebra y cálculo relacional* como lenguaje para hacer las consultas permite que no haya ambigüedad. En otro tipo de bases de datos como la de red sí es posible tener ambigüedad. La precisión en las consultas incrementa el alcance de la mismas. En otros modelos de bases de datos estas pueden volverse muy complejas, lo cual limita las capacidades de las mismas.

Independencia: la estructura normalizada usada en una BDR permite tener más fácilmente independencia de los datos que en Bases de Datos tipo red, por ejemplo.

Seguridad: es fácil añadir permisos de acceso a variables de interés únicamente añadiendo una nueva relación con los controles de autorización deseados.

Desventajas

Rendimiento: Si la consulta deseada usa muchas tablas o el tamaño de las mismas es grande la query puede tardar un tiempo considerable en correr.

Costo: Para la construcción de la base y su mantenimiento se requiere software especializado y/o programador para construir la base y posteriormente un administrador de la misma.

Seguridad: La estructura de BDR tiene poca seguridad. Si se requiere seguridad adicional se deberá destinar recursos adicionales con este fin.

Límites del tamaño: Algunas BDR tienen límites en el tamaño del campo que deben ser especificados cuando se diseña la misma.

No escalable: Las BDR organizan la información por características comunes. Imágenes complejas, números y productos multimedia no son complejos y no son fácilmente procesables por una BDR, se procesan con una BD orientada a objetos.

Aislamiento: al almacenar grandes cantidades de información, las BDR se convierten en "grandes islas de información" y pueden no compartirse fácilmente con otras. La comunicación entre BDR suele ser costosa y tardada.

Lenta realización de un "*quick analysis*": A menos que la base de datos esté organizada y guardada de manera jerárquica es tardado hacer análisis rápidos de la información.

Bodegas de Datos, BD (dataware house, DHW)

Una Bodega de Datos (BD) es una herramienta que permite que los usuarios finales realicen fácilmente consultas sobre sus datos sin necesidad de acceder a la operación del sistema con el objeto de **tomar mejores decisiones** respecto a una situación específica.

Una BD se crea, sencillamente, cuando datos de sistemas anteriores son copiados en un nuevo sistema dedicado exclusivamente a analizarlos. Así, se separa el proceso automatizado de obtener los datos y vaciarlos en bases, del proceso de manejarlos. Estos datos quedan a disponibilidad de todos los usuarios, para entender mejor qué pasó y qué está pasando, y con esto tomar mejores decisiones.

Los datos contenidos en la BD (también llamadas DWH, por sus siglas en inglés) pueden venir de una o varias fuentes; las BD proporcionan un robustez y flexibilidad para combinar diferentes fuentes de datos en un sistema combinado eficiente.

Elementos que la integran

Los elementos básicos de una BD son, esencialmente:

1. Sistemas Operativos - es el servicio que captura los datos. En realidad, se encuentra “fuera” de la BD, pues se tiene nulo o poco control sobre el formato de los datos. Su propósito es que la información esté disponible. En la generalidad, no son utilizados para consultas de datos. Mantienen pocos datos históricos.
2. Sistema de Extracción-Transformación-Carga (también llamado ETL, por sus siglas en inglés) - consiste en un área de trabajo, estructuras de datos y un conjunto de procesos. Su función es leer los datos de todos los sistemas operativos (extracción), transformar, limpiar, combinar diferentes fuentes y quitar duplicados (transformación) y, por último, estructurar y cargar los datos en los modelos de presentación (carga).
3. Herramientas de Presentación de Datos - en ella se organizan, guardan y preparan para consultas directas por los usuarios. Es la única parte alcanzable para las Herramientas de Acceso a Datos. Es en este nivel en el que existe el diseño dimensional de la Bodega, de la cual se hablará en la siguiente sección.
4. Herramientas de Acceso a Datos - estas pueden ser tan simples como una herramienta de consulta básica (consultas directas de SQL) o tan complejas como un sistema de minería de datos o aplicaciones de modelos. En este nivel, es posible que existan interfaces para que los usuarios no necesiten crear sus propias herramientas de consulta, o incluso se pueden desarrollar herramientas de modelado o predicción que guarden sus resultados en cualquiera de los otros tres niveles.

Diseño de Bodegas de Datos

El diseño dimensional es la base de todas las BD actuales. Hay dos productos principales del diseño dimensional: esquema de estrella (también conocido como bases de datos relacionales) y cubo.

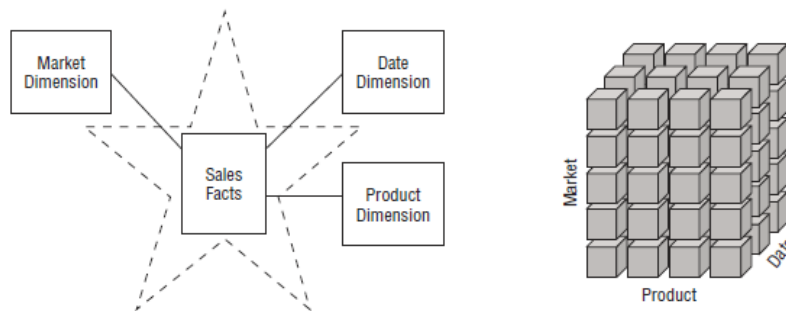


Figure 1-1: Star schema versus OLAP cube.

El primer acercamiento al sistema de datos comienza por búsquedas directas: una orden individual, una medición en específico, etc. Sin embargo, las necesidades de análisis pronto se vuelven más complejas: queremos comparar las ventas de este mes con las ventas de este mismo mes el año pasado (como se hace en negocios) , o comparar miles de mediciones de un fenómeno para poder acercarnos al valor real (como se hace en secuenciación). Para este tipo de comparaciones, es necesario establecer procesos que no solamente proporcionan información individual, sino con el procesamiento general.

De esta manera, el modelado dimensional aborda los requerimientos únicos de sistemas analíticos. El diseño dimensional optimiza el acceso a grandes cantidades de datos. Además, permite el mantenimiento de los datos históricos guardados en la BD, incluso cuando nuevos sistemas operativos cambien o borren información.



Cubos de información

Los modelos dimensionales no siempre están implementados en BDR. Una base de datos multidimensional guarda información en un formato llamado *cubo*. Básicamente, un cubo se basa en preestablecer combinaciones de dimensiones, para que después puedan ser accedidas y estudiadas interactivamente.

Un cubo permite, principalmente, cambiar la perspectiva que se tiene de los datos, por medio de añadir o remover atributos, y recibir resultados instantáneamente. A este proceso se le suele llamar Procesamiento Analítico en Línea (OLAP, por sus siglas en inglés).

Además, las bases de datos multidimensionales están libres de muchas de las limitaciones de SQL. Este tipo de bases comenzó a ofrecer totales, categorización y ordenación, y operaciones estadísticas mucho antes de que estas capacidades fueran añadidas a SQL. Además, dado que están construidas específicamente para trabajar con dimensiones, las bases de datos multidimensionales proporcionan herramientas para generar jerarquías recursivas.

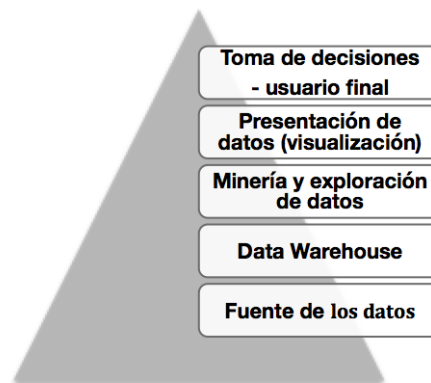
Sin embargo, el escalamiento es muy limitado en los cubos. A medida que las dimensiones y la cantidad de valores en ellas crecen, el número de combinaciones que pueden ser preestablecidas explota. Los cubos, entonces, están muy limitados para trabajar con grandes volúmenes de datos. Además, la accesibilidad a las APIs multidimensionales no está tan difundida: es mucho más común el conocimiento de SQL.

	Data Structure	Access Language	Style of Interaction	Advantages
Relational Database	 Star Schema	Structured Query Language (SQL)	Query and response	Scalable Widely understood access language
Multi-dimensional Database	 Cube	Proprietary API or MDX	Interactive (OLAP)	Fast Expressive access language

Las bases de datos relacionales y los cubos funcionan bien juntas. Por ejemplo, la BDR puede servir como la bodega de datos, usando cubos como una capa adicional para tener reportes de alto desempeño. Tal vez, en un futuro cercano, sea posible esquemas relacionales que sean guardados como cubos, para acceder a ellos con un lenguaje sencillo (SQL), pero con respuestas interactivas y rápidas, como OLAP.

Funciones del almacenamiento de datos

La BD es la reproducción de datos coordinada, estructurada y periódica (batch-oriented processing) que proviene de varias fuentes al interior y fuera de la organización en cuestión. La reproducción se realiza en un ambiente óptimo para el proceso analítico e informado de los datos. El propósito principal de las BD es almacenar en un sistema la información de múltiples aplicaciones dentro y entre las organizaciones. Estos grupos de información son transformados en temas específicos para permitir consultas, análisis, reportes, entre otros de manera accesible y consistente para asistir en la **toma de decisiones**.



En general, las BD tienen tres funciones básicas:

- 1) **Validación:** validar los datos que ya se creían ciertos.
- 2) **Reportes:** uso de los datos de las BD para reportes tácticos.
- 3) **Exploración:** exploración de nuevas ideas.

Este último punto se reconoce como la actividad en donde las BD tiene mayor potencial; no obstante, la mayor las funciones sulene concentrarse en los reportes tácticos.

Por otro lado, es necesario que las BD cumpla algunos requisitos que permitirán sacar provecho de la información que éste contiene. La organización de la información debe ser **accesible**, de manera que los contenidos sean claros e intuitivos para el usuario. Además es necesario que la información presentada sea **consistente**, por ello la información contenida debe ser previamente limpiada y analizada para garantizar su calidad. Asimismo, se sugiere que la BD se pueda adaptar a las necesidades de los usuarios finales, sin afectar o cambiar los datos existentes. Estas condiciones ayudarán a las BD a cumplir su función principal: servir como base para mejores tomas de decisiones.

Ventajas

Uno de los puntos clave de las BD es que rompe con las barreras creadas por tener la información separada y desligada, que se traducen en falta de información en la toma de decisiones.

- Centralización de los datos.
- Permite administrar y organizar los datos de acuerdo con las necesidades de los usuarios.
- Facilita el proceso de toma de decisiones.
- Facilita el acceso a información de distintas fuentes, un sistema común de almacenamiento se traduce en análisis y reportes más sencillos.
- Acceso a consultas de la información sin alto grado de apoyo técnico.
- Particularmente útil en el mediano y largo plazo.
- Las BD permite transformar información bruta en útil.

Para poder tener los beneficios de las BD es necesario tomar en cuenta los siguientes aspectos:

1. Establecer el objetivo de las BD en el proyecto.
2. Indicar la función específica que se requiere de las BD.
3. Establecer qué datos debe contener la BD para poder cumplir sus funciones específicas.
4. Establecer que tan grande será la BD.

Estos puntos ayudarán a hacer eficiente la BD, descartando información innecesaria y evitando coleccionar información "sin contenido".

Desventajas

Las desventajas principales están relacionadas con el tiempo que se debe dedicar a la preparación de la información a considerar. La información debe ser extraída de las fuentes de origen, es necesario limpiarla y cargarla al sistema lo cual puede consumir gran cantidad de tiempo. Este proceso generalmente se automatiza por completo; no obstante, requiere mantenimiento continuo.

- Dificultad para añadir nuevas fuentes de datos.
- Diseño complejo que requiere de múltiples disciplinas, lo cual puede resultar en una BD sin utilidad.
- Posibles problemas con la compatibilidad de los datos.
- Dificultad en su mantenimiento.
- Tiempo de implementación largo, que se refleja en altos costos.
- No es la mejor herramienta para basar la toma de decisiones en tiempo real, ya que puede requerir tiempos largos de procesamiento (las BD en tiempo real están probando ser muy útiles- e-commerce).

Importancia

La información se adquiere de múltiples fuentes y es almacenada en el mismo sitio, lo cual permite cruzar la información y organizarla de manera conveniente para proveer soluciones que no serían posibles con los datos separados. La exploración y análisis de los datos de manera conjunta permite descubrir posibles patrones y conexiones que favorecen la toma de decisiones. La minería de datos puede dar soporte clave para aprovechar la información contenida en una BD.

Las empresas que deciden usar BD es porque ven que la información a nivel “departamental” puede tener valor estratégico cuando es integrada.

Mineros

Un minero es aquel software que nos provee de las herramientas necesarias para realizar minería de datos. Tenemos varios mineros importantes:

Software	Desarrollador	Lenguaje	Plataforma	Licencia
Orange	University of Ljubljana	Java	Windows Linux Mac OSx	GNU General Public License
WEKA	University of Waikato	Python		
R	GNU Project	C y Fortran		

Orange

Características:

- Preprocesamiento de datos
- Modelado
- Técnicas de exploración
- Visualización
- Recuerda decisiones pasadas y da sugerencias
- Add-ons para bioinformática
- Tiene componentes para Aprendizaje de Máquina
- Interfaz gráfica de usuario o python scripts

WEKA

Características:

- Preprocesamiento de datos
- Clasificación
- Regresión
- Clustering

- Visualización
- Acceso a Bases de Datos SQL usando Java Database Connectivity (JDBC)
- Interfaz gráfica de usuario o línea de comandos

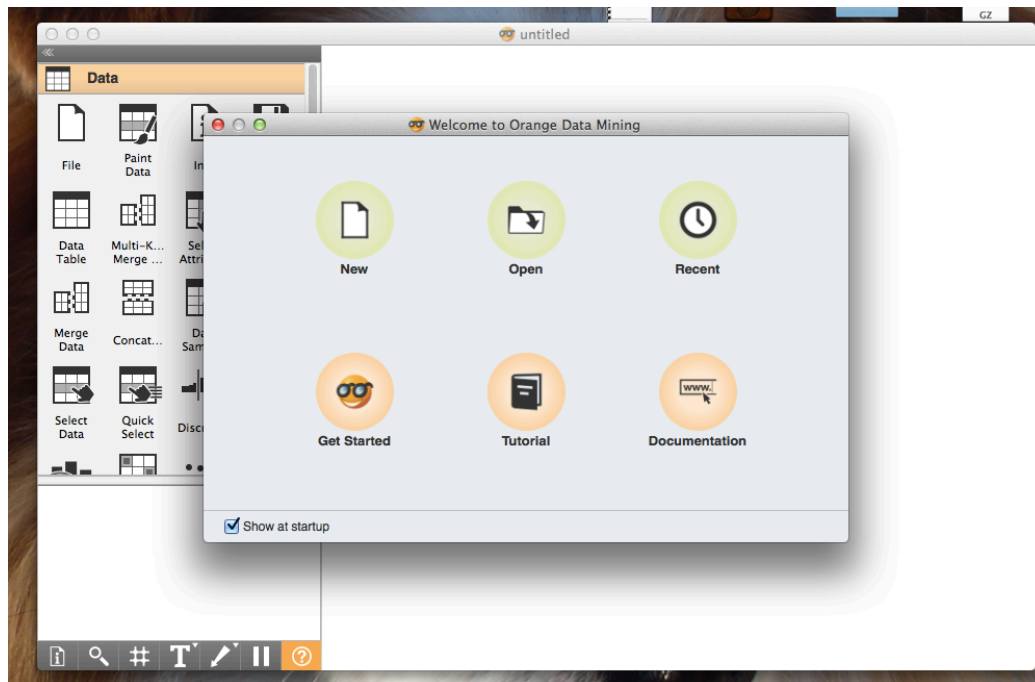
R

Características:

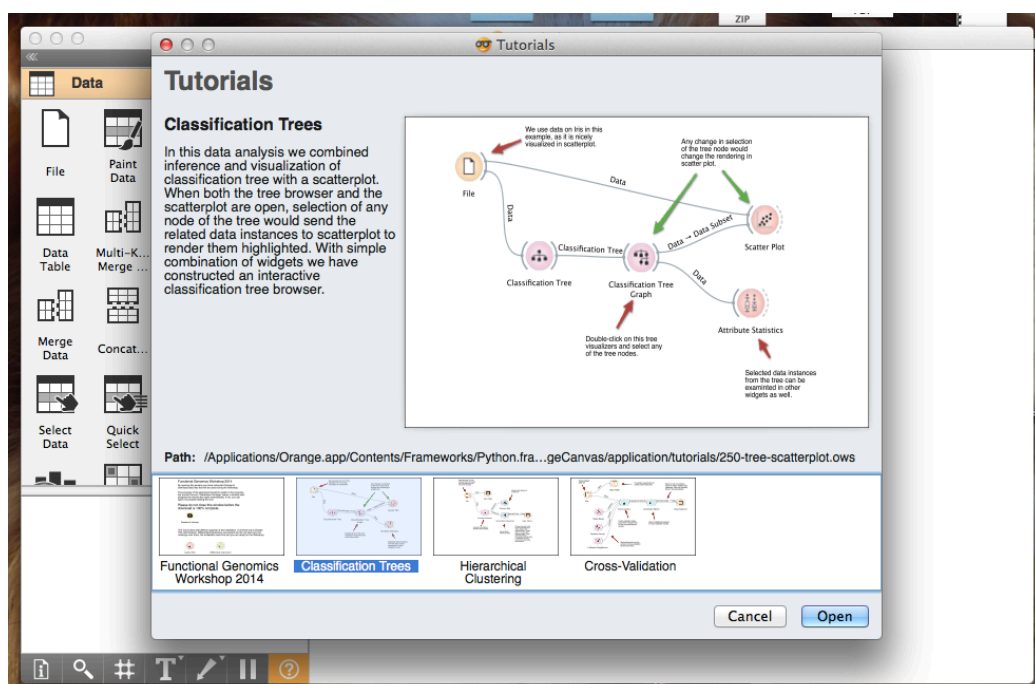
- Es un Lenguaje y un ambiente de programación para computación estadística
- Permite realizar Modelado Lineal y no Lineal
- Contiene paquetes para clustering, aprendizaje de máquina, text mining, entre otros.

ANEXO. EJEMPLO DE USO DE ORANGE

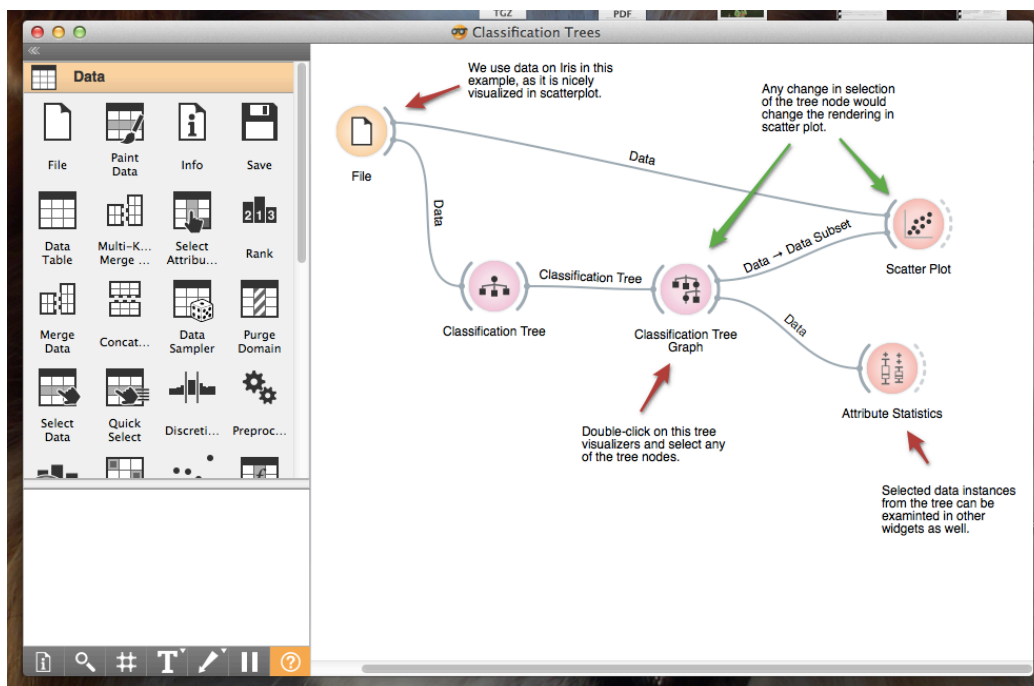
- Pantalla de Inicio.



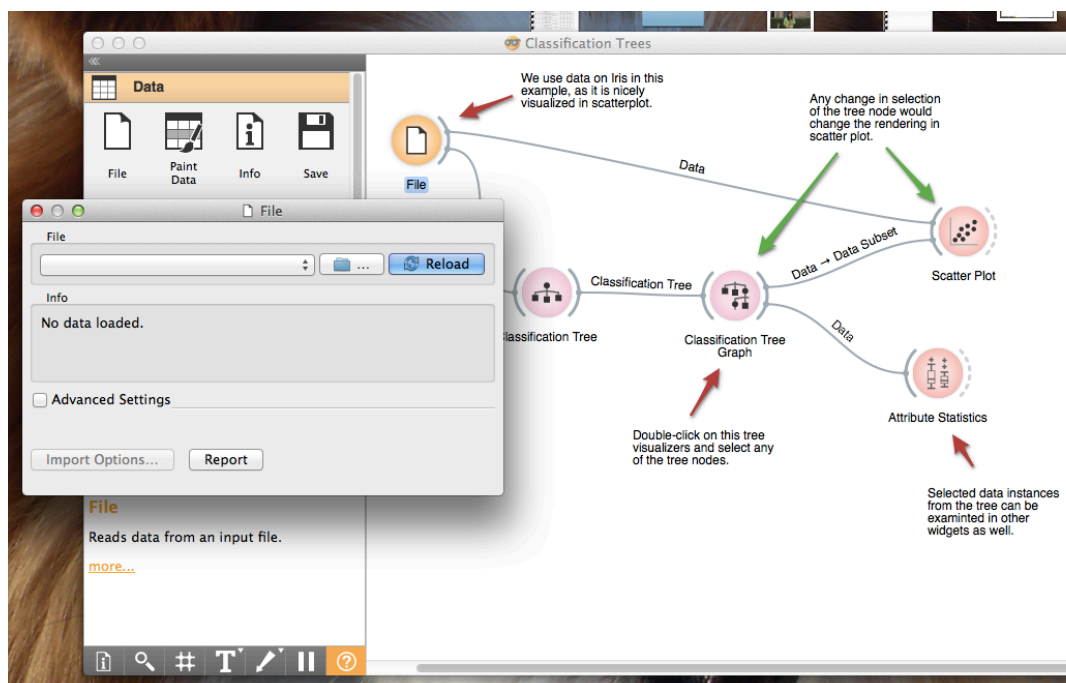
- Para ejemplificar usaremos como ejemplo uno de los tutoriales que ya contiene Orange.



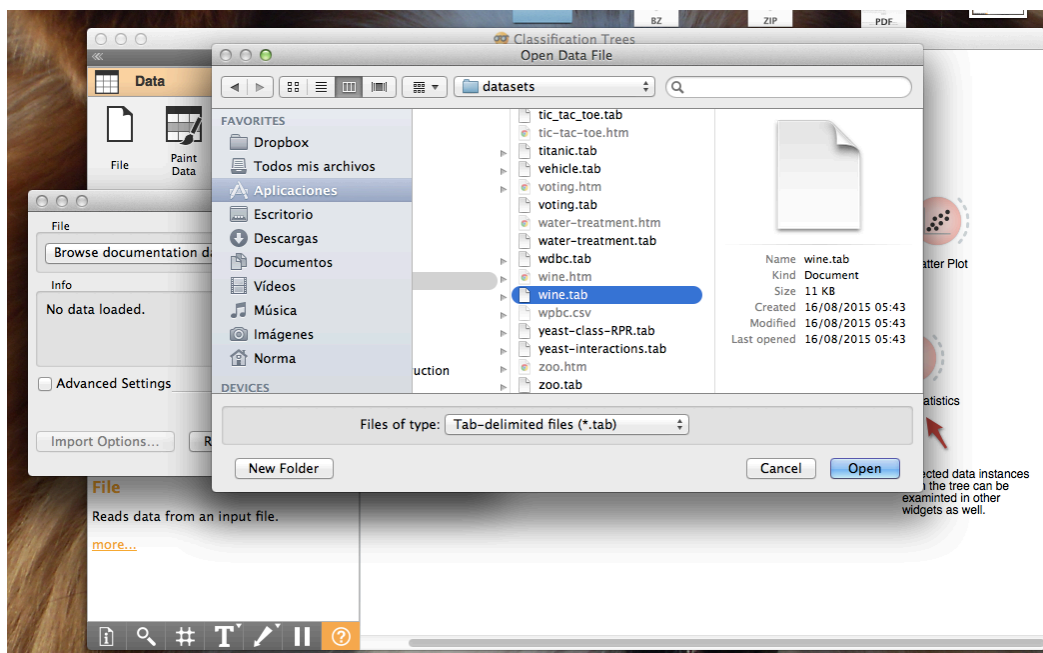
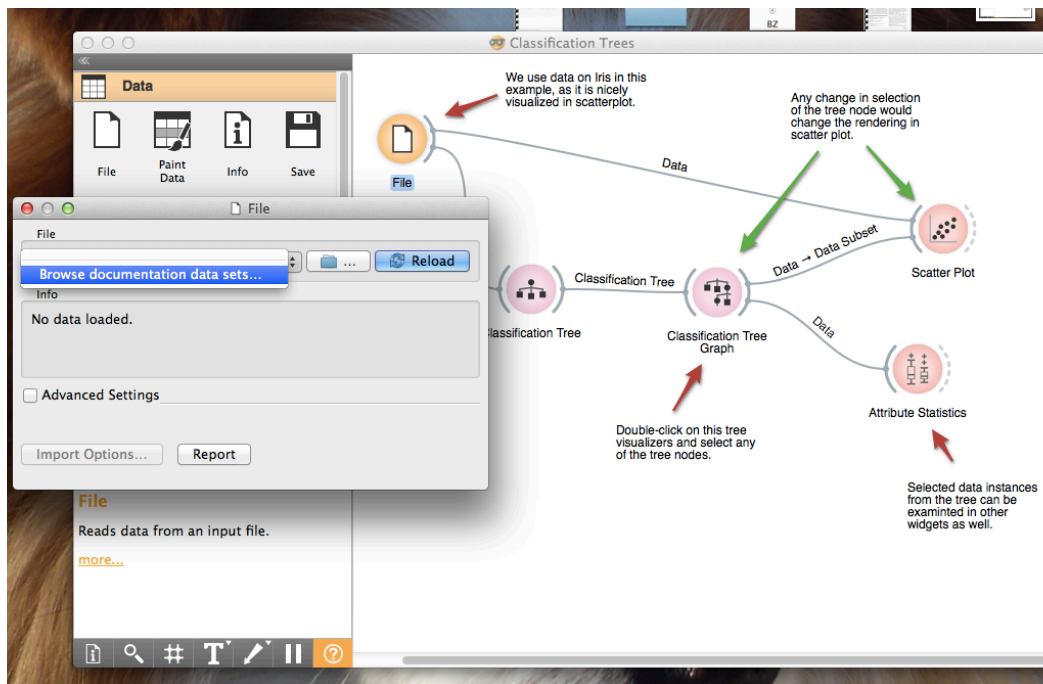
- Vista.



- Para seleccionar nuestro archivo de entrada, basta con dar doble click en la imagen File.

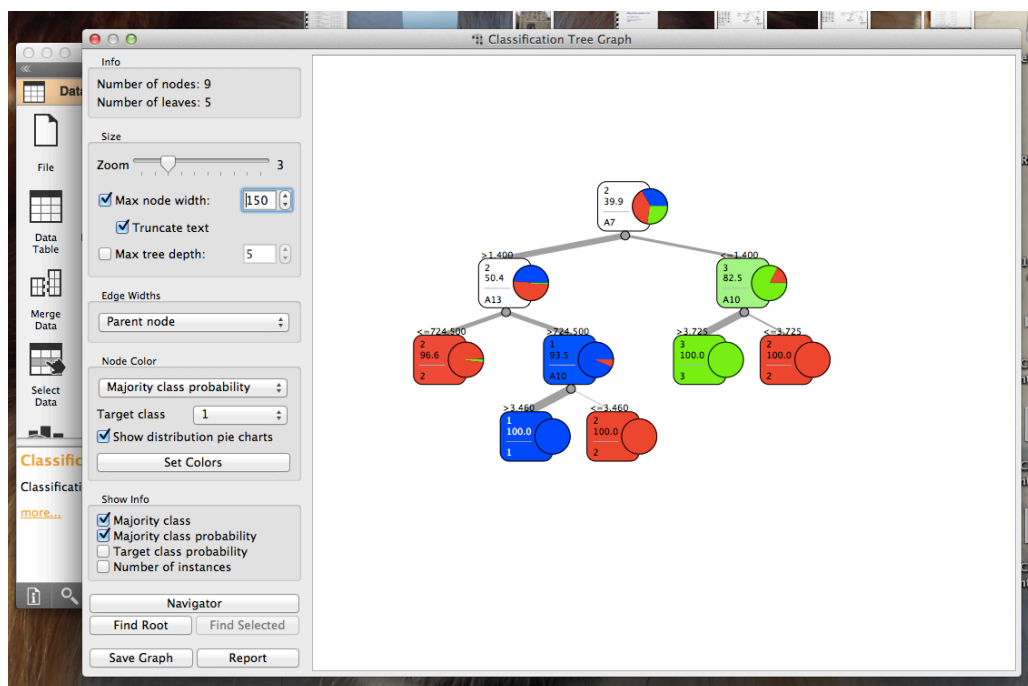
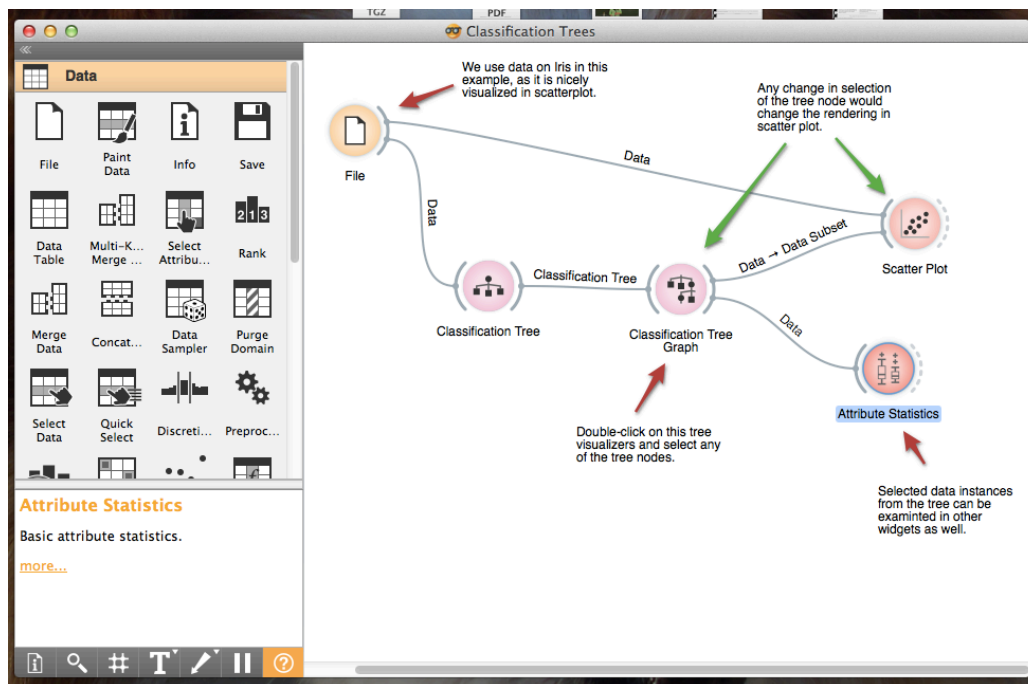


- Seleccionar el archivo.

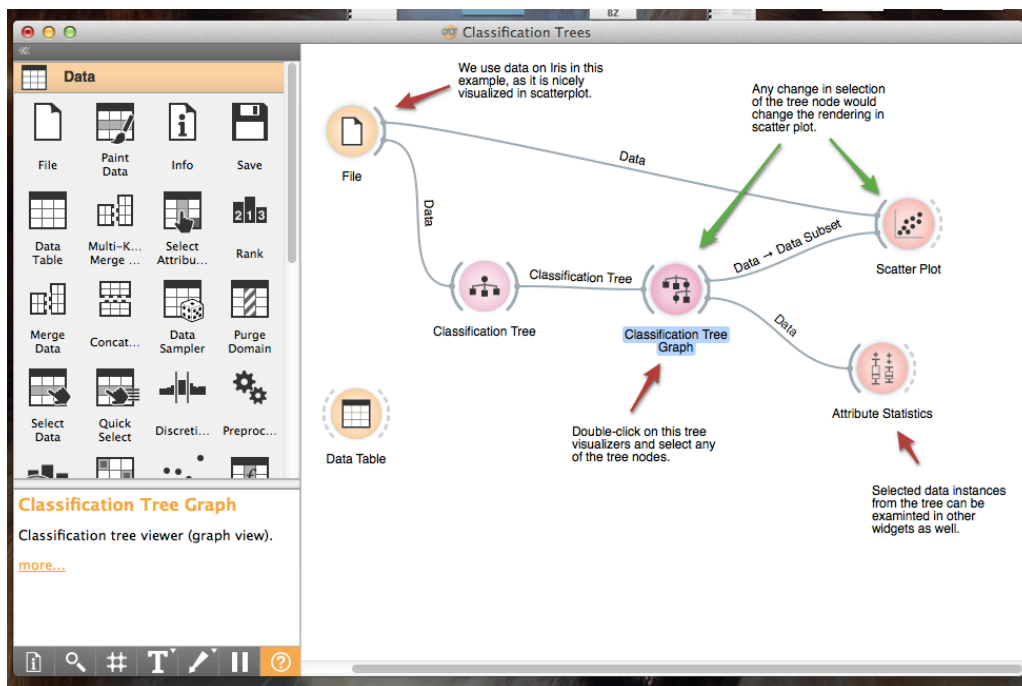


- Una vez seleccionado el archivo presionamos el botón reload para cargarlo.
- Orange actualizará cada paso de nuestro esquema con los datos de entrada del archivo.

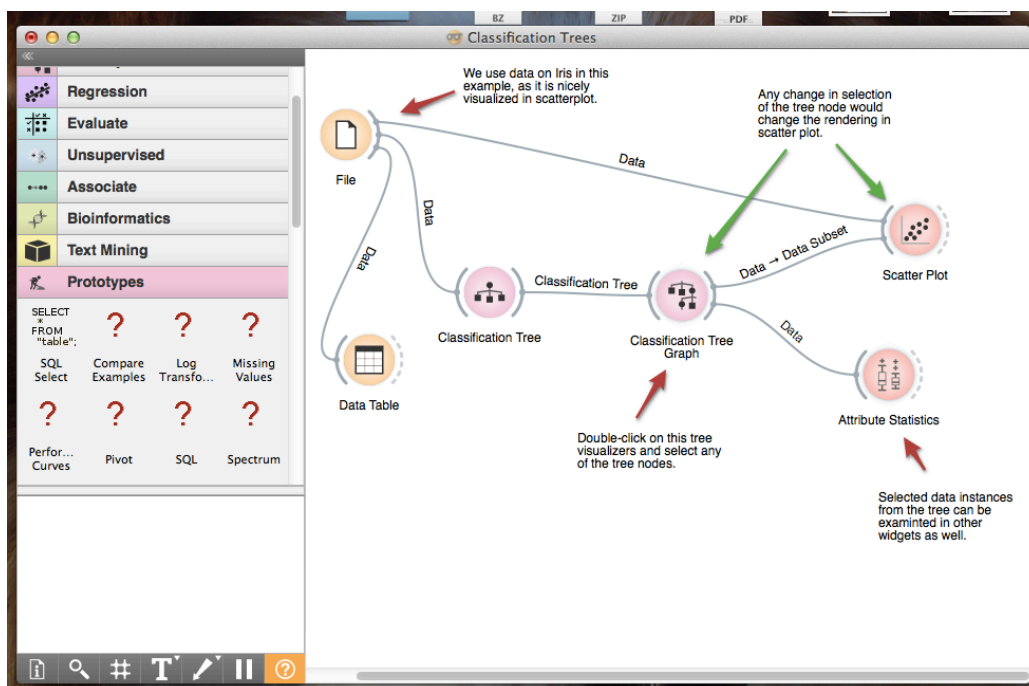
- Para ver el resultado de alguna etapa, basta con dar doble click sobre la imagen de ésta.



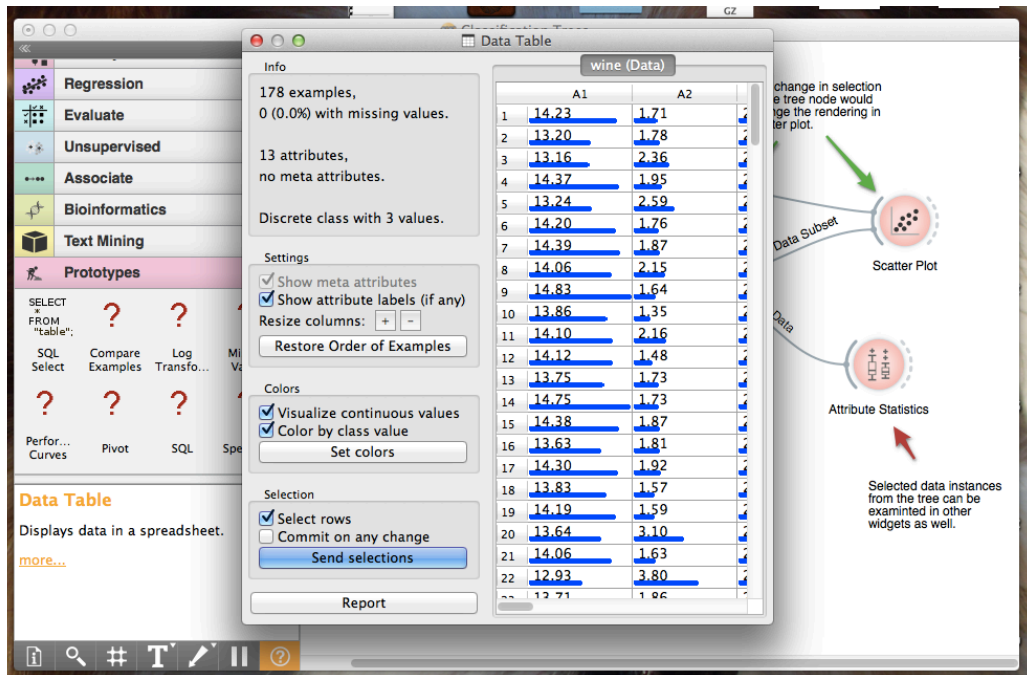
- Si deseamos agregar un nuevo componente, lo seleccionamos del menú que se encuentra a la izquierda y lo arrastramos a donde queremos colocarlo, en este caso añadimos Data Table.



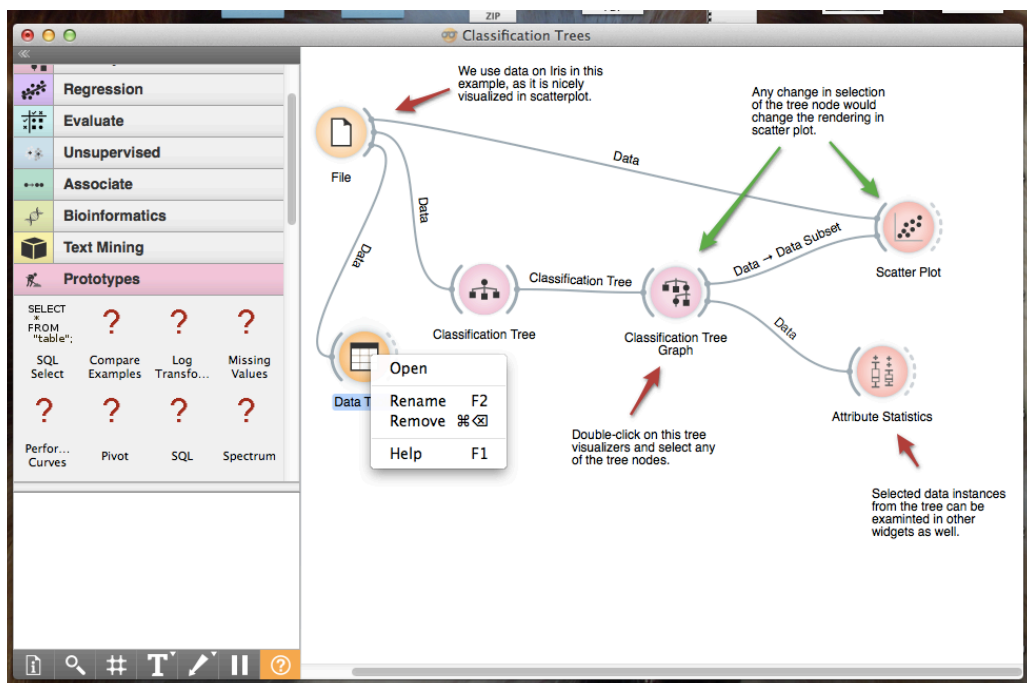
- Para relacionar dos elementos, damos doble click sobre la línea punteada de alguno de los elementos y después damos click sobre el elemento con el que lo queremos relacionar, la línea se pintará.



- Para actualizar los resultados al nuevo componente, seleccionamos la imagen File y presionamos el botón Reload, posteriormente si damos click sobre el componente recientemente añadido veremos los cambios.



- Para eliminar algún componente, damos click sobre él con el botón secundario y elegimos la opción Remove.



Bibliografía

Ramakrishnan, Gehrke. (2002) *Database Management Systems*, Third edition, McGRAW Hill

Abraham Silberschatz , Henry F. Korth , S. Sudarshan. (2011) *Database System Concepts*, Sixth Edition, McGRAW Hill

C.J. Date. (2001) *Introducción a los Sistemas de Base de Datos*, 7a Edición, Alhambra Mexicana

Kurt Windisch; Louis Davidson; Scott Klein; Kevin Kline. (2008) *Pro SQL Server 2008 Relational Database Design and Implementation*, First Edition, Apress

Witten, I y Frank, E. (2005) *Data Mining: Practical Machine Learning Tools and Techniques* ELSEVIER, Estados Unidos de América.

Kimball, R. y Ross, M. (2002) *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling* Wiley Computer Publishing, Estados Unidos de América.

Academic Tutorials (2008) <http://www.academictutorials.com/> [Revisado el 15 de agosto de 2015].

The Complete Reference Star Schema, Christopher Adamson, McGraw Hill, 2010
Data Warehousing: Using the Wal-Mart Model, Paul Westerman, Academic Press, 2001, USA

The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling, Ralph Kimball and Margy Ross, Wiley, 2013, USA

40 top free Data Mining software <http://www.predictiveanalyticstoday.com/top-free-data-mining-software/> [Revisado el 15 de agosto de 2015]

5 of the best free and Open Source Data Mining software
<http://www.junauza.com/2010/11/free-data-mining-software.html> [Revisado el 15 de agosto de 2015]

Six of the best Open Source Data Mining tools <http://thenewstack.io/six-of-the-best-open-source-data-mining-tools/> [Revisado el 15 de agosto de 2015]

Orange <http://orange.biolab.si/> [Revisado el 17 de agosto de 2015]

Machine Learning Group at the University of Waikato
<http://www.cs.waikato.ac.nz/ml/index.html> [Revisado el 18 de agosto de 2015]

R Project <https://www.r-project.org/> [Revisado el 18 de agosto de 2015]

Relational databases <http://maxlogix.blogspot.mx/2009/09/advantages-and-disadvantages-of-using.html> [Revisado el 18 de agosto de 2015]

J.G. Zheng (2012) *Relational Database Basics Review*
<http://jackzheng.net/teaching/it4153/files/database-basics-review.pdf>

POSTGRESQL'S TRANSACTION MODEL

<https://www.packtpub.com/books/content/postgresqls-transaction-model> [Revisado el 18 de agosto de 2015]

POSTGRESQL Table Expressions 9.3

<http://www.postgresql.org/docs/9.3/static/queries-table-expressions.html> [Revisado el 18 de agosto de 2015]