



Ku Information System

Referencia Técnica

Guía para la implementación de sistemas de información

Qué es un sistema de información Ku?

Un SI es una base de datos normalizada que integra la información que proviene de otras bases de datos. El SI contiene únicamente datos relevantes para la consulta, categorizados de forma homogénea y optimizados para el acceso de consulta.

En un SI la información se divide en **áreas**, que normalmente se refieren a distintos aspectos de la actividad (pérdidas y ganancias, ventas, objetivos, cash flow, etc). En esta documentación, las áreas frecuentemente se denominan también **contextos** de información.

Contextos

La información del contexto consiste únicamente en variables que representan valores numéricos (ventas, índices, importes, distancias,...). Estos valores numéricos están clasificados por un conjunto finito de criterios (clientes, productos, edades, meses, ...). Esta estructura, conocida como *hipercubo*, permite acceder a las variables (la información) mediante cualquier combinación de criterios (los ejes de consulta). En resumidas cuentas, tenemos que

$$\text{contexto} = \text{criterios} + \text{variables}$$

En Ku un contexto (el hipercubo) se almacena en una tabla de base de datos que agrupa los criterios y variables.

Criterios

Los criterios pueden ser de varios tipos.

Los más frecuentes son los criterios tipo **atributo**, y representan cualquier cosa que se puede enumerar con una lista. El usuario puede seleccionar elementos de esta lista, que son pares de clave y título. Cada criterio de este tipo se almacena en una tabla separada (normalmente en tablas simples tipo clave/título), y el conjunto de tablas de criterios se denomina **diccionario**. A nivel de base de datos, existe una relación de uno a muchos entre cada tabla de criterio y cada tabla de contexto que lo utilice.

Otro tipo de criterios son los de tipo **periodo** y **fecha**, que representan el tiempo. Se almacenan de un modo que permite fácilmente extraer o agrupar la información a distintos niveles, y el usuario puede seleccionar rangos.

El último tipo de criterio disponible en Ku es el tipo **void**, que representa cualquier otra cosa. No están referenciados en ninguna parte, ni pueden seleccionarse. Lo único que puede hacerse con ellos es verlos en una consulta. Este tipo de criterio se usa para datos muy variables y detallados que no es necesario incluir en el diccionario. Por ejemplo, números de factura o de póliza.

Variables

Las variables son siempre numéricas, pero pueden tener diferentes características:

Pueden ser **acumulables** o no por las funciones de totales y/o subtotales de Ku. La mayoría de variables lo son, pero hay excepciones, como por ejemplo variables que representen precios, saldos, etc. No tiene sentido sumar precios de distintos productos, pero sí lo tiene sumar importes de venta.

Pueden tener **dependencia temporal**. Un caso claro de variable que depende del tiempo es el “stock”. No podemos mostrar el “stock de todo un año” como una suma de los stocks del mes, porque lo que tiene sentido en todo caso sería el “stock al final del año”. Cuando una variable se define como temporal, Ku sólo permitirá al usuario utilizarla en la consulta si todos los criterios tipo DATE o PERIOD del contexto cumplen al menos uno de los siguientes supuestos:

- la consulta lo incluye en cualquiera de sus ejes, a último nivel (día en un DATE, mes en un PERIOD).
- la consulta lo incluye como filtro, a último nivel, designando un único periodo.

En resumen, si definimos una variable como temporal, estamos obligando al usuario a utilizarla teniendo en cuenta esta dimensión. Podríamos dejar esto a criterio del usuario, pero eso podría producir consultas que llevaran a confusión.

Finalmente una variable también pueden ser **moneda**, cuando representa valores monetarios que pueden convertirse. En ese caso el usuario tendrá posibilidad de ver esos valores convertidos a una moneda distinta.

Cada variable define el número de posiciones decimales con las que se representará en las consultas.

Además de las variables almacenadas en las tablas de los contextos, Ku permite definir variables calculadas a partir de constantes, variables almacenadas u otras variables calculadas previamente.

Cómo se define un sistema de información Ku

Diseñar el SI

El primer paso consiste en identificar qué contextos va a incluir el SI. Para cada contexto, se identificarán las variables a incluir (los datos que necesitará el usuario), y los criterios por los que se pueden clasificar (los ejes que utilizará para representar los datos). Nótese que más de un contexto puede compartir el mismo criterio: el diccionario es global para todos los contextos del SI.

Definir un nombre para cada contexto, criterio. Los nombres de contextos y criterios deben ser únicos en todo el SI. A estos nombres se les conoce como **id de criterio** o **id de contexto**. Es recomendable que sean cortos, por ejemplo, identificadores de 3 o 4 caracteres. También daremos un nombre a cada *variable*, aunque estos no tienen que ser únicos en el SI, basta con que lo sean en cada contexto.

Crear el almacén de datos

Crear una base de datos en el servidor de elección. En esta base de datos, crear una tabla para cada contexto, y una tabla para cada criterio de tipo atributo. Deben usarse los mismos **ids de criterio** y **contexto** consistentemente como **nombres de tabla** y **campo**, respectivamente. La denominación y definición de los campos debe tener en cuenta las especificaciones de la sección *Tipos de criterios*.

Crear el diccionario

El diccionario Ku siempre es local, y debe ser una base de datos MS Access 97. Por tanto, *crear una base de datos Access, importar desde el almacén de datos todas las tablas de criterios* y verificar que cada tabla tiene su correspondiente **Primary Key**. Si se usa una versión reciente de Access, utilizar la opción “Convertir a la versión anterior de MS Access” para guardar el diccionario.

Crear el archivo de definición de esquema

Por último, *describir el SI en un archivo de esquema*. Aquí se enumerarán los criterios y contextos, y para cada contexto, se especificarán sus variables y otros atributos. Véase la sección *Formato del archivo de esquema del sistema de información*.

Escribir los procesos de importación

Una vez definido el SI, debe llenarse con información. Estos procesos deben extraer los datos del origen e insertarlos debidamente en el almacén de datos. Estos procesos deben cumplir determinados requisitos, en especial para posibilitar la replicación automática del SI. Véase la sección *Procesos de importación*.

Ejemplo de SI

Haremos un ejemplo para demostrar todo el proceso. Supongamos un SI muy sencillo (rematadamente sencillo) de análisis de ventas.

Diseñar el SI

Nos bastará con un único contexto, que llamaremos “Ventas”. Le damos el identificador “VEN”. Las variables que nos interesan son el importe de venta, el importe de coste de la venta, y el margen tanto en importe como en porcentaje. Queremos poder analizar la información por cliente, producto, zona y mes de venta. Comprobamos que la información puede obtenerse de las fuentes de datos disponibles, y definimos el contexto, que queda más o menos así :

criterios :

CLI Cliente, atributo

PRD Producto, atributo

ZON Zona, atributo

F Periodo de venta, tipo periodo.

contextos :

VEN Ventas

criterios utilizados:

CLI, PRD, ZON, F

variables:

V Importe de venta, moneda, acumulable

C Importe a coste de la venta, moneda, acumulable

M Margen (variable calculada = $V-C$), moneda, acumulable

PM Porcentaje margen (variable calculada = $(M*100)/V$), no acumulable

Crear el almacén de datos

Con la información del diseño, ya podemos crear el almacén de datos. Creamos (en un DBMS cualquiera) una base de datos SIVENTAS, donde crearemos las siguientes tablas:

Tablas de criterios:

Tabla CLI clientes

CLI carácter(20), clave primaria

TXT carácter(50)

Tabla PRD productos

PRD carácter(20), clave primaria

TXT carácter(50)

Tabla ZON zonas de venta

ZON carácter(20), clave primaria

TXT carácter(50)

Tablas de contextos:

Tabla VEN contexto de ventas

CLI carácter(20)

PRD carácter(20)

ZON carácter(20)

FY entero, periodo de venta (año)

FM entero, periodo de venta (mes)

FH entero, periodo de venta (semestre)

FQ entero, periodo de venta (trimestre)

V decimal, importe venta

C decimal, importe compra

La tabla VEN no necesita clave primaria, pero de tenerla debería incluir todos los campos excepto las variables. Para acelerar los accesos, haremos un índice separado por cada uno de los criterios de tabla, y tres índices para el criterio de periodo : FY+FM, FY+FH, FY+FQ. Todos estos índices son aceleradores facultativos que pueden usarse a discreción, pero no son imprescindibles. De todas formas, si los recursos no son un problema (y la flexibilidad de implantación de Ku contribuye a que no lo sea), se recomienda crearlos.

También crearemos una fuente de datos ODBC asociada a esta base de datos recién creada. La llamaremos SIVENTAS.

Tablas para el replicador:

Tendremos que crear siempre un par de tablas que utilizará el replicador para controlar las actualizaciones de datos. Son las siguientes.

Tabla ZZREPCTL

Columnas:

ID char(20), PK
LASTSEQ char(14)

Tabla ZZREPLOG

Columnas:

ID char(20), PK
SEQ char(14), PK
SELECT char(250)

Crear el diccionario

Crearemos una base de datos Access 97, la llamaremos SIVENTAS.MDB e importaremos de nuestra base de datos SIVENTAS recién creada todas las tablas. En realidad, SIVENTAS.MDB va a ser una réplica de SIVENTAS que puede utilizarse como datawarehouse (por ejemplo, para su uso en un portátil offline).

Crear el archivo de definición de esquema

Crearemos un archivo ascii llamado SIVENTAS.TXT y definimos el SI como sigue:

```
[KRIT]
CRIT1=DBSMALL,CLI,Cliente,047
CRIT2=DBSMALL,PRD,Producto,138
CRIT3=DBSMALL,ZON,Zona,116
CRIT4=PERIOD,F,Periodo venta,27

[CTX1]
ID=VEN
NAME=Ventas
ICON=48
CRIT=CLI,PRD,ZON,F
VAR1=V,sum(V),Importe venta,21,Y,2,1,0
VAR2=C,sum(C),Venta a coste,22,Y,2,1,0
VAR3=M,=V-C,Margen,24,Y,2,1,0
VAR4=MP,=(M*100)/V,% margen,34,N,2,0
```


Procesos de extracción

La función de los procesos de extracción es, sencillamente, alimentar el datawarehouse, tomando los datos de donde quiera que estuviesen. Si los datos en origen son similares a lo que espera el SI, la extracción puede ser casi una mera copia. Si hay que obtener los datos de distintas fuentes y sincronizarlos puede llegar a resultar un proceso de complejidad considerable. Por tanto, en principio no son procesos complejos, a menos de que sea necesario también **transformar** la información antes de incorporarla. La complejidad dependerá de la diferencia que haya entre lo que se espera obtener y lo que se tiene para conseguirlo.

El proceso de extracción debe notificar al replicador qué conjunto de datos y criterios está extrayendo. Para ello, grabará la información pertinente en los archivos ZZREPCTL y ZZREPLOG después de cada extracción.

El proceso de extracción también asume la responsabilidad de saber qué es lo que introduce en el datawarehouse. Es responsabilidad del extractor saber, cuando procesa, por ejemplo, ventas de 2003/4, qué debe ocurrir con las ventas del mismo período que pudiera haber ya en el datawarehouse: las hay? Debe sustituirlas porque esto es un refresco? Debe simplemente añadir las nuevas porque es una extracción incremental?

Esto depende del propósito del contexto en cuestión. Hay contextos que se reemplazan completos cada día, hay otros a los que se les van incorporando acumulados meses enteros, etc. En cualquier caso, es el extractor quien debe garantizar que la información que deja en el datawarehouse es consistente.

Una práctica habitual y razonable es definir los conjuntos por periodos, y reemplazar siempre grupos completos. Cuando se añade información de un periodo al datawarehouse, previamente se elimina este mismo periodo del datawarehouse. Ku y el replicador hacen esto mediante conjuntos definidos en SQL. En cada extracción, se introduce en el datawarehouse un conjunto de datos que se pueda definir mediante una cláusula SQL where (que podría ser nula). Con esto se puede, por ejemplo, delimitar el período 2003/4 (“FY=2003 and FM=04”).

Antes de incorporar los datos, el proceso de extracción elimina el citado conjunto del datawarehouse (con una sentencia SQL delete con esa misma cláusula). El resultado es que la extracción sustituye un “conjunto de datos”, en vez de simplemente apilarlos.

A continuación el proceso de extracción inserta esa misma cláusula en el archivo de control del replicador. Cuando el replicador quiera replicar el datawarehouse en otro servidor, antes de incorporar el periodo 2003/4 hará un borrado del mismo periodo en el datawarehouse destino, utilizando la cláusula where suministrada por el extractor.

Proceso de extracción de ejemplo

Supongamos un proceso de extracción para el contexto VEN del ejemplo anterior. Supongamos, además, que hemos decidido actualizar el datawarehouse por meses completos. Veamos algo de pseudocódigo informal para un proceso así:

construir una \$cadenaSQL como "FY=año and FM=mes"

eliminar el período del datawarehouse
(delete from datawarehouse.VEN where \$cadenaSQL)

obtener los datos y asignar filas para el contexto VEN
para cada fila
 para cada criterio de tabla en VEN (en este caso, CLI, PRD, ZON)
 si el criterio no existe en la tabla de criterios correspondiente (CLI, PRD, ZON)
 añadirlo a dicha tabla
 recordar : criterio modificado
 añadir la fila a la tabla VEN

construir un \$numeroSecuencia como "aaaammddhhmmss" con la hora del sistema

para cada contexto o criterio que haya tenido al menos una adición (en este caso, VEN, CLI, PRD o ZON)
 insertar en datawarehouse.ZZREPCTL una fila con los valores:
 ID=id del criterio o contexto
 LASTSEQ=\$numeroSecuencia

insertar en datawarehouse.ZZREPLOG una fila con los valores:
 ID="VEN"
 SEQ=\$numeroSecuencia
 SELEKT=\$cadenaSQL

Este código asegurará que el replicador se entere de los cambios efectuados al datawarehouse, y asegurará que este datawarehouse (y todos los que se repliquen de él) mantendrá la consistencia por periodos.

Otro aspecto importante del extractor es el volumen de información que genera. Es muy recomendable que la información en el datawarehouse esté sumariada al último nivel, ya que el detalle más allá del último nivel no aporta nada al usuario y aumentará la carga de procesador y el espacio en disco sin ninguna ventaja. Una posibilidad de asegurar esto es (si el DBMS lo permite) especificar una **clave primaria** (única) para la **tabla de contexto** que incluya **todas** las columnas que son **criterios**.

En el ejemplo, el código

obtener los datos y asignar filas para el contexto VEN
para cada fila

debería generar una única fila en VEN para cada combinación de CLI,PRD,ZON,FY y FM.

Por último, **en ningún caso el proceso de extracción debe alterar información en el datawarehouse que no esté incluida en el conjunto definido (el que se registra en ZZREPLOG)**. El replicador no se enterará de que otro conjunto ha sido alterado, y los almacenes dependientes quedarán en estado inconsistente.

Tipos de criterios

Atributo (DBSMALL)

Criterio tipo código/descriptivo. Este tipo de criterio requiere una tabla que contenga los pares de valores. En las tablas de contexto de información, se almacena únicamente la clave.

Selecciones posibles :

- Todo
- Valores individuales seleccionados por el usuario

Representaciones posibles :

- **clave, título** o ambos, en cualquier orden.

Identificador en el archivo de esquema :

- **DBSMALL**

Requisitos en el almacén de datos : valores de criterios

- El nombre de la tabla ha de coincidir con el **id de criterio**
- Debe contener al menos 2 campos :
 - **clave**, campo de carácter. El nombre de campo ha de coincidir con el **id de criterio**
 - **título**, campo de carácter. El nombre de campo ha de ser "**txt**"
- Debe tener un índice primario sobre el campo de **clave**

Requisitos en el almacén de datos : contexto

- Cada contexto que lo utilice ha de contener :
 - Un campo con el valor de la **clave**. El nombre de campo ha de coincidir con el **id de criterio**

Periodo (PERIOD)

Este criterio básicamente expresa **meses**. Es un criterio jerárquico de hasta 2 niveles : el primero siempre es el año. El segundo, opcional, será mes, trimestre o semestre.

Selecciones posibles :

- Todo
- Periodo actual (según se seleccione año, mes, trimestre o semestre.)
- Rango desde hasta (idem)

Representaciones como

- Año
- Año / mes
- Año / trimestre
- Año / semestre

Identificador en el archivo de esquema :

- **PERIOD**

Requisitos en el almacén de datos : contexto

- Cada contexto que lo utilice ha de contener, suponiendo que el **id de criterio** es XXX :
 - XXXy año
 - XXXm mes
 - XXXh semestre
 - XXXq trimestre
- Se recomiendan los índices : y, y+m, y+q, y+h

Fecha (DATE)

Este criterio es una ampliación de PERIODO, llegando hasta el nivel **día**. Es un criterio jerárquico de hasta 3 niveles.

Selecciones posibles :

- Todo
- Periodo actual (según se seleccione año, mes, trimestre o semestre, semana, etc.)
- Rango desde hasta (idem)

Representaciones como

- Año
- Año / mes
- Año / mes / día
- Año / trimestre
- Año / semestre
- Año / semana
- Año / semana / día de la semana

Identificador en el archivo de esquema :

- **DATE**

Requisitos en el almacén de datos : tabla de contexto

- Cada contexto que lo utilice ha de contener, suponiendo que el **id de criterio** es XXX :
 - XXXy año
 - XXXm mes
 - XXXh semestre
 - XXXq trimestre
 - XXXwsemana
 - XXXd día del mes
 - XXXx día de la semana
- Se recomiendan los índices : y, ym, yq, yh, ymd, ywd

Formato del archivo de esquema del sistema de información

El esquema del sistema de información se almacena en un archivo ASCII de propiedades (tipo .ini). Estos archivos consisten básicamente en secciones, especificadas así:

[SECCION]

y que a continuación definen una serie de pares de clave/valor, así:

CLAVE=VALOR

El sistema de información se define mediante una sección [KRIT], que enumera todos los criterios del SI, y una sección [CTXn] para cada contexto “n” o área de información.

Criterios – Sección [KRIT]

Todos los criterios del SI deben especificarse con una clave CRITn para cada criterio, siendo “n” un número de secuencia. No debe haber saltos en el número de secuencia, pero pueden aparecer en cualquier orden. El valor de la clave será la definición del criterio, indicando la clase de criterio, el identificador, un texto descriptivo y un número de icono. El identificador debe coincidir con el nombre de campo del criterio en el diccionario y la base de datos.

[KRIT]

CRITn=**tipo**,**id**,**descripción**,**icono**

Donde ...

tipo = DBSMALL, PERIOD, DATE, VOID

id = identificador del criterio

descripción = descripción del criterio

icono = nº de icono 1-167, véase sección *Iconos para criterios y contextos*

Contextos de información – Sección [CTXn]

Todos los contextos (áreas de información) del SI deben especificarse con una sección CTXn, siendo “n” un número de secuencia. No debe haber saltos en el número de secuencia, pero pueden aparecer en cualquier orden. Dentro de esa sección se especificarán las siguientes claves:

[CTXn]

ID = identificador del contexto, debe coincidir con el nombre de tabla en la BD

NAME = texto corto descriptivo del contexto

ICON = nº de icono 1-167, véase sección *Iconos para criterios y contextos*

CRIT = lista de criterios utilizados por este contexto, separada por comas

A continuación, existirá una clave VARn para cada variable “n” definida en el contexto, siendo “n” un número de secuencia. No debe haber saltos en el número de secuencia, pero pueden aparecer en cualquier orden. La clave VARn definirá la variable con una lista de propiedades, separada por comas, como sigue:

VARn=id,expresion,titulo,icono,acumulable,dec,moneda,dec2,temporal

Donde:

id

identificador de la variable. Si no es una variable calculada, debe coincidir con el nombre de campo correspondiente en la base de datos.

expresion

expresión que obtiene o calcula el valor de la variable. La expresión puede ser un fragmento SQL, tal cual, o una expresión aritmética, precedida por el signo “=”.

Será un fragmento de expresión SQL cuando el dato se obtiene directamente de un campo de la base de datos, y este fragmento se utilizará tal cual para componer las consultas. Como las consultas siempre obtienen los datos sumariados por algún criterio, la expresión sql será casi siempre algo como **sum(nombre_campo)**, aunque puede incluir cualquier otra función agregada o aritmética soportada por SQL. Por ejemplo:

VAR7=CV, sum(CV), venta a coste, 22, Y, 2, 1, 0

Será una expresión aritmética cuando el dato se obtiene mediante cálculo a partir de otras variables. En ese caso la expresión debe ir precedida por el signo “=”, y puede hacer referencia a constantes o a cualquier variable que ya haya sido definida en este contexto (es decir, cuya clave sea VARx donde x<n), ya sea de base de datos o calculada. El analizador aritmético admite todos los operadores habituales y el uso de paréntesis. Por ejemplo:

VAR10=PMA, =(IVN-CV)*100/IVN, % margen, 34, N, 2, 0

titulo

texto corto descriptivo del contexto

icono

nº de icono "TC". Véase la sección *Iconos para variables*

acumulable

define si la variable es acumulable **por las funciones de totales y/o subtotales de Ku**. La mayoría de variables lo son, pero hay excepciones, como por ejemplo variables que representen precios, saldos, etc.

"Y": la variable es acumulable

"N": la variable no es acumulable. Ku mostrará los subtotales en blanco.

dec

define las posiciones decimales con las que se representará la variable en Ku.

moneda, dec2

estos dos valores definen si la variable es un valor monetario que puede convertirse.

"moneda" puede ser:

0 = la variable no representa un valor monetario

1 = la variable es un valor monetario en moneda **local**

2 = la variable es un valor monetario en moneda **alternativa**

"dec2" define las posiciones decimales en moneda alternativa.

temporal

T = define si la variable depende del tiempo

Iconos para criterios y contextos

La colección de iconos asociados a criterios o contextos es fija y consta de 167 iconos. La muestra de estos iconos se incluye en la carpeta **ku\ikons** a modo de referencia. Los nombres de estos iconos empiezan por "x" y van numerados de "x001.ico" a "x167.ico". Nótese que cambiar las imágenes de los iconos de esta carpeta no tendrá efecto en lo que represente Ku, ya que en la versión actual estos iconos están incluidos en el ejecutable KU.EXE.

El icono se especifica en la definición de un criterio o contexto mediante un identificador numérico "999" en el rango 0..167, que ha de corresponder con alguno de los iconos en **ku\ikons**. Si no se especifica, o se especifica un identificador inválido, el criterio o contexto adoptarán un icono por omisión.

Por ejemplo, la siguiente línea asocia el ikono nº 131 al criterio **fam**. La imagen que se representará es la de `\ku\ikons\x131.ico` :

```
CRIT5=DBSMALL , fam , Familia , 131
```

El siguiente ejemplo especifica el icono nº 48, esta vez para representar el contexto **fac** :

```
[CTX1]  
ID=FAC  
NAME=Facturación  
ICON=48
```

Iconos para variables

La colección de iconos asociados a variables es fija, y está orientada a representar el **tipo de dato**, más que el contenido. El contenido suele ser demasiado variado o abstracto como para disponer de una colección de iconos adecuada. Así, hay unos iconos base para representar datos tipo *unidades*, *importes*, *porcentajes* y *precios*. Estos iconos base están disponibles en colores *negro*, *rojo*, *verde* y *azul*, para hacer más intuitiva la identificación (por ejemplo, gastos y pérdidas pueden ir en rojo, márgenes y beneficios en verde, estimaciones en azul, etc.)

El icono se especifica en la línea de definición de una variable, mediante un identificador "TC" de dos dígitos, siendo "T"=Tipo y "C"=Color. Si no se especifica, o se especifica un identificador inválido, la variable no mostrará icono alguno. Los tipos y colores disponibles son :

Iconos de variables : tipos de datos

- | | |
|---|-------------|
| 1 | Unidades |
| 2 | Importes |
| 3 | Porcentajes |
| 4 | Precios |

Iconos de variables : colores

- | | |
|---|-------|
| 1 | Negro |
| 2 | Rojo |
| 3 | Azul |
| 4 | Verde |

Así, por ejemplo, el identificador de icono **24** producirá un icono tipo **importe** de color **verde**, como en la línea siguiente :

VAR8=IMA , sum(IVN-COS) , Margen , 24 , N