

Artículo

[Ricardo Paiva](#) · Dic 10, 2021 Lectura de 4 min

## Recomendaciones para almacenar grandes conjuntos de datos

Me gustaría compartir algunas funciones de almacenamiento que también existen en Caché y que son prácticamente desconocidas y en su mayoría no se utilizan. Por supuesto, están disponibles en IRIS y son más relevantes con arquitecturas de almacenamiento extensas y distribuidas.

Cuando defines una clase persistente en tu Intercambio electrónico de datos (EDI) o una tabla usando DDL, normalmente seleccionas un nombre significativo que se relaciona con el contenido y el contexto de su tabla. Ya sabes que los datos y los índices se almacenan en globals y no debes preocuparte por ello. El compilador realiza la definición basándose en el nombre de la clase o de la tabla. Con excepción de los nombres de las clases de más de 29 caracteres, puedes predecir los nombres de los globals estándar con bastante facilidad, como se describe en la [documentación](#).

Class dc.MyCompany.EmployeeRegister Extends %Persistent  
Storage Default

```
<DataLocation>^dc.MyCompany.EmployeeRegisterD</DataLocation>
<IdLocation>^dc.MyCompany.EmployeeRegisterD</IdLocation>
<IndexLocation>^dc.MyCompany.EmployeeRegisterI</IndexLocation>
<StreamLocation>^dc.MyCompany.EmployeeRegisterS</StreamLocation>
```

Esto es bastante cómodo y te permite concentrarte en la lógica de su solución. Prácticamente no tiene impacto en el rendimiento, ya que los excelentes métodos de almacenamiento dentro de las bases de datos utilizan algoritmos de compresión muy eficientes. Fuera de la base de datos tenemos varias partes que tienen largos nombres de globals: Primero, el Journal en el almacenamiento del disco.

Después, cualquier función que transporte nombres de globals en algunas redes como: ECP puro, Mirroring, Sharding. Como la clasificación de la velocidad no ha cambiado con todas las mejoras tecnológicas puede emplear la siguiente dirección:

Memoria > Almacenamiento > Red > Usuario.

Así que con un tráfico de red intensivo (Sharding, ECP) o Journal+Red (Mirror), hay algunas desventajas.

Para tener el control sobre los nombres de los globals, está el parámetro [DEFAULTGLOBAL](#). Al aplicar esto al ejemplo anterior se obtiene este resultado:

*/// Set the root of the global names*

```
Parameter DEFAULTGLOBAL = "^dc.er";
```

Storage Default

```
<DataLocation>^dc.erD</DataLocation>
<IdLocation>^dc.erD</IdLocation>
<IndexLocation>^dc.erI</IndexLocation>
<StreamLocation>^dc.erS</StreamLocation>
```

Esto es mucho mejor, pero no es lo mejor que puedes conseguir. Como sabes, con este enfoque cada índice se almacena en el primer nivel del subíndice de <IndexLocation>.

Si tu clase/tabla tiene muchos índices, este primer nivel podría ser significativo. Esta mezcla no es atractiva si tienes una combinación de

- índices clave puros
- índices recopilados

- índices con datos
  - índices de mapa de bits & índices bitslice
- Sería más eficiente tener cada índice en su propio global

La solución para dividir los índices en globals individuales es el parámetro [USEEXTENTSET](#). Si ampliamos el ejemplo anterior, veremos este resultado:

```
Parameter DEFAULTGLOBAL = "^dc.er";
Parameter USEEXTENTSET = 1;
Storage Default
<DataLocation>^dc.er.1</DataLocation>
<ExtentLocation>^dc.er</ExtentLocation>
<IdLocation>^dc.er.1</IdLocation>
<Index name="IDKEY"><Location>^dc.er.1</Location></Index>
<Index name="xDOB"><Location>^dc.er.2</Location></Index>
<Index name="xName"><Location>^dc.er.3</Location></Index>
<IndexLocation>^dc.er.l</IndexLocation>
<StreamLocation>^dc.er.S</StreamLocation>
```

Ahora cada índice tiene una estructura de almacenamiento homogénea. Si omites el parámetro DEFAULTGLOBAL, obtienes nombres globales. La configuración predeterminada si ejecutas Sharding será la siguiente. En este caso:

```
Parameter USEEXTENTSET = 1;
Storage Default
<DataLocation>^BLWQ.C069.1</DataLocation>
<ExtentLocation>^BLWQ.C069</ExtentLocation>
<IdLocation>^BLWQ.C069.1</IdLocation>
<Index name="IDKEY"><Location>^BLWQ.C069.1</Location></Index>
<Index name="xDOB"><Location>^BLWQ.C069.2</Location></Index>
<Index name="xName"><Location>^BLWQ.C069.3</Location></Index>
<IndexLocation>^BLWQ.C069.l</IndexLocation>
<StreamLocation>^BLWQ.C069.S</StreamLocation>
```

Comentarios finales:

Los efectos de los 2 parámetros se reconocerán únicamente con conjuntos de datos de gran tamaño y aparecen especialmente en cualquier instalación de Sharding.

Encontrarás más información en esta documentación

[Hashed Global Names](#) , [Index Global Names](#) , [Table Definition Optimization](#)

[#Globals](#) [#Modelo de datos de objetos](#) [#Rendimiento](#) [#Tablas relacionales](#) [#Caché](#) [#Ensemble](#) [#HealthShare](#)  
[#InterSystems IRIS](#) [#InterSystems IRIS Analytics \(DeepSee\)](#) [#InterSystems IRIS for Health](#)

URL de fuente: <https://es.community.intersystems.com/post/recomendaciones-para-almacenar-grandes-conjuntos-de-datos>