

NoSQL: Key-Value

Gerardo Rossel



DEPARTAMENTO
DE COMPUTACION

2017

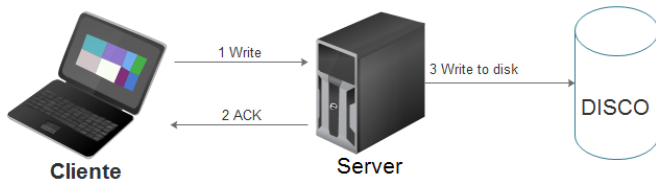
Key-Value

Origen

- Dynamo-Amazon
 - Giuseppe DeCandia, et al., Dynamo: **Amazon's highly available key-value store**. In Proceedings of twentyfirst ACM SIGOPS symposium on Operating systems principles (SOSP '07). ACM, New York, NY, USA,

Key-Value

- Diccionario o array asociativo
- Key-Value:
 - Namespaces o Buckets
 - In-Memory vs On-Disk
- Características
 - Simples
 - Escalables
 - Veloces.



¿Claves?

- Bases relacionales:
 - Garantizar la inmutabilidad de la la clave primaria.
 - Se usan claves sin significado.
- Key-Value
 - No hay columnas, no hay manera de saber el significado de un valor excepto dándole semántica a la clave.
Cart[12387] = 'SKUAK8912j4'
CustName[12387] = 'Katherine Smith'

¿Cómo construir una clave?



¿Cómo construir una clave?

Entity Name + ':' + Entity Identifier + ':' + Entity Attribute

- *Cliente : 12345678 : Apellido*
- *Producto : SKU110 : Nombre*
- Dependiendo de la BD hay soporte para varios tipos en los valores.
 - Redis soporta valores de: Strings, Lists, Sets, Sorted sets, Hashes, Bit Arrays
 - Keys en Redis son *binary safe*
- **Ojo:** las claves sirven también para organizar valores en múltiples servers

Key-Value: Keys

- Usar nombres significativos y no ambiguos

Key-Value: Keys

- Usar nombres significativos y no ambiguos
- Usar partes basadas en rango si se necesita recuperar rangos de valores (enteros, fechas)

Key-Value: Keys

- Usar nombres significativos y no ambiguos
- Usar partes basadas en rango si se necesita recuperar rangos de valores (enteros, fechas)
- Usar un delimitador común ":"

Key-Value: Keys

- Usar nombres significativos y no ambiguos
- Usar partes basadas en rango si se necesita recuperar rangos de valores (enteros, fechas)
- Usar un delimitador común ":"
- Mantener las claves lo más cortas posibles sin sacrificar las otras características.

```
define getCustAttr(p_id , p_attrName)
    v_key = 'cust'+':' +p_id+':' +p_attrName;
    return (AppNameSpace[ v_key ] );

define setCustAttr(p_id ,p_attrName ,p_value)
    v_key = 'cust'+':' +p_id+':' +p_attrName;
    AppNameSpace[ v_key]=p_value
```

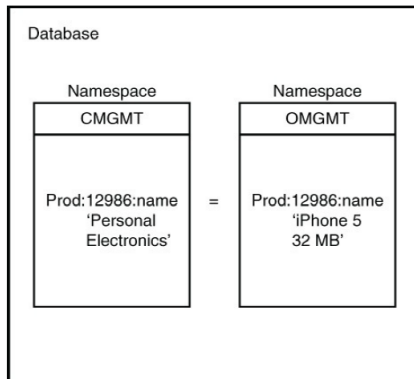
¿Cómo guardar Valores?

- String: '1232 NE River Ave, St. Louis, MO'
- Lista: ('1232 NE River Ave', 'St. Louis', 'MO')
- HASH: **Street** '1232 NE River Ave' **City** 'St. Louis' **State** 'MO'
- JSON:

```
{ "Street" : "1232 NE River Ave", "City" : "St. Louis", "  
  State" : "MO" }
```








Espacio de Nombres

Espacios de nombre permiten evitar conflictos



Tiempo de Vida

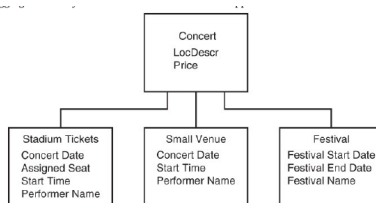
Venta de tickets. Guardar asientos mientras se procesa la compra.

Key	TTL
'U7138'	
R3194	
S2241	
T1294	
K4111	
R1143	
S1914	



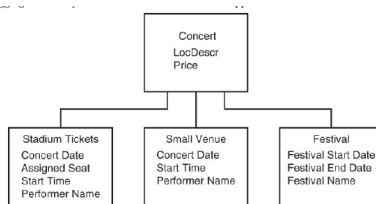
Agregados

- ¿Como almacenar agregados?



Agregados

• ¿Como almacenar agregados?



```
{ "type": "stadium", "conDate": 15-Mar-2015, "
  locDescr": "Springfield Civic Center", "
  assgnSeat": "J38", "startTime": "17:30", "
  price": "$50.00", "perfName": "National" }
```

```
{ "type": "small venue", "conDate": 12-Jun-2015,
  locDescr": "Plymoth Concert Hall", "
  startTime": "17:30", "price": "$75.00", "
  perfName": "Joshua Redman" }
```

```
{ "type": "small venue", "conDate": 12-Jun-2015, locDescr": "Plymoth Concert Hall", "startTime":
  "17:30", "price": "$75.00", "perfName": "Joshua Redman" }
```

Agregados Atómicos

```
ConcertApp[ticketLog:9888] = {"conDate":15-Mar-2015, "locDescr": "Springfield Civic Center",  
    "assgnSeat": "J38"}
```

Vs.

```
ConcertApp[ticketLog:9888:conDate] = 15-Mar-2015  
ConcertApp[ticketLog:9888:locDescr] = "Springfield Civic Center"  
ConcertApp[ticketLog:9888:assgnSeat] = "J38"
```

Indices

```

define addLocAssgnSeat(p_locDescr, p_seat)
begin
v_seatList = ConcertApp[p_locDescr]
v_seatList = append(v_seatList, p_seat)
ConcertApp[p_locDescr] = v_seatList
end;

```

Users

UserID	Info
7734	City:San Francisco, email:alef@gmail.com
4667	City:New York, email:jsample@yahoo.com
6578	City:Seattle, email:knovoselic@gmail.com
1263	City:San Francisco, email:jeray@yahoo.com

Cities

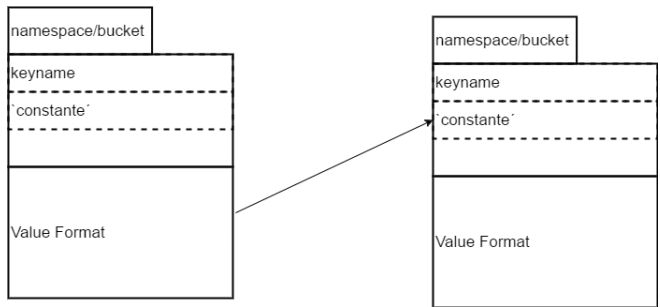
City	UserIDs
San Francisco	7734, 1263, ...
New York	4667, ...
Seattle	6578, ...

Indices

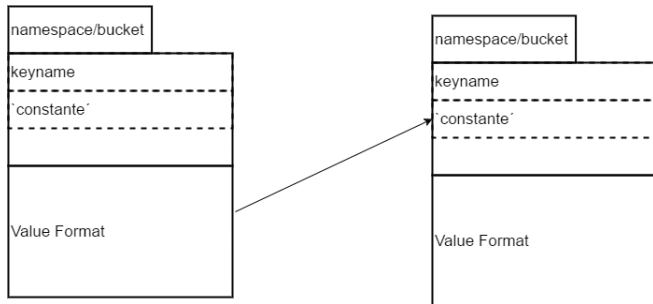
Muchas bases de datos key-value soportan índices. En caso contrario se pueden usar índices invertidos.

Notación Key-Value

- Espacio de nombres optativo
- Partes de la clave
- Valor en formatos JSON, Lista, Hash, string, etc.
- Relación entre un valor y una clave de otro dato.

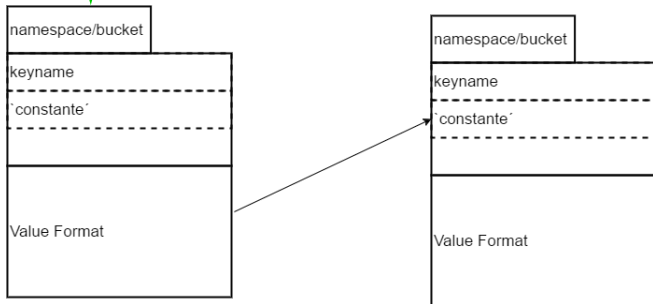


Notación Key-Value



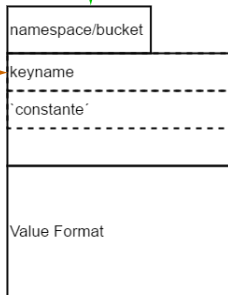
Notación Key-Value

Espacio de nombres optativo

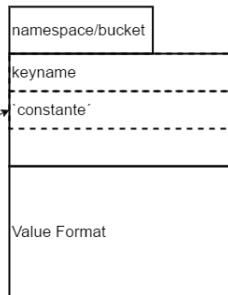


Notación Key-Value

Espacio de nombres optativo

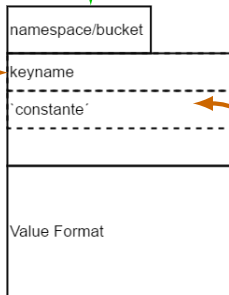


Parte variable de la clave



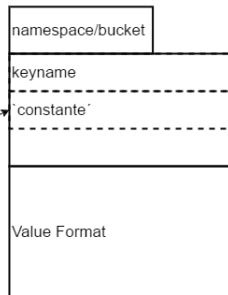
Notación Key-Value

Espacio de nombres optativo



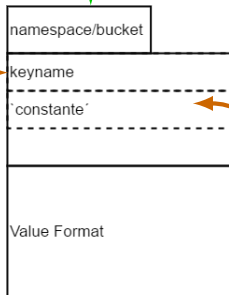
Parte variable de la clave

Parte constante (semántica) de la clave



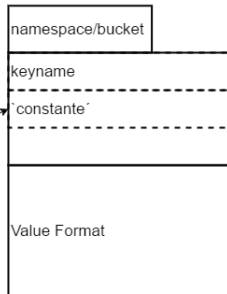
Notación Key-Value

Espacio de nombres optativo



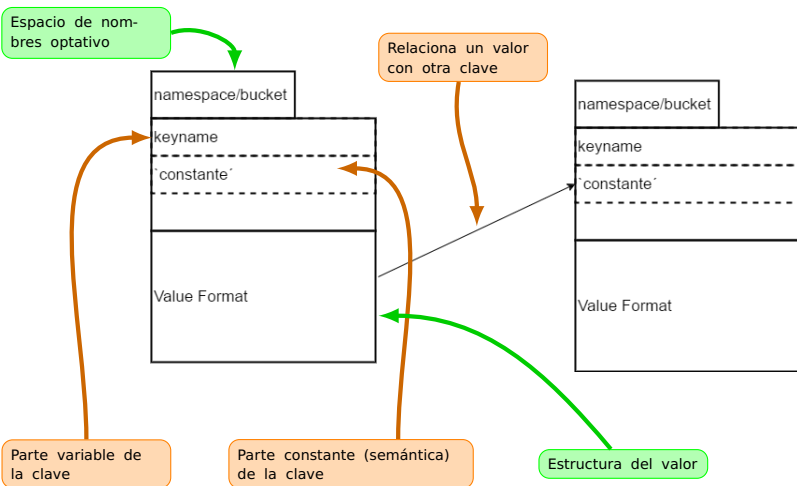
Parte variable de la clave

Parte constante (semántica) de la clave



Estructura del valor

Notación Key-Value

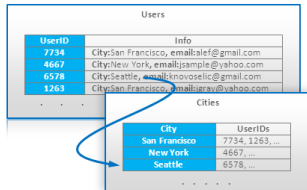


Notación Key-Value Ejemplo

```
ConcertApp[ticketLog:9888] = {"conDate":15-Mar-2015, "locDescr": "Springfield Civic Center",
                             "assgnSeat": "J38"}
```

ConsertApp	ConsertApp	ConsertApp
'ticketlog'	'ticketlog'	'ticketlog'
nroticket: int	nroticket: int	nroticket: int
JSON: 'assgnSeat': {"type": "string"}, 'conDate': {"type": "string", "format": "date - time "}, 'locDescr': {"type": "string"}	HASH assignSeat: string conDate: date locDescr: string	JSON: Schema ConsertApp

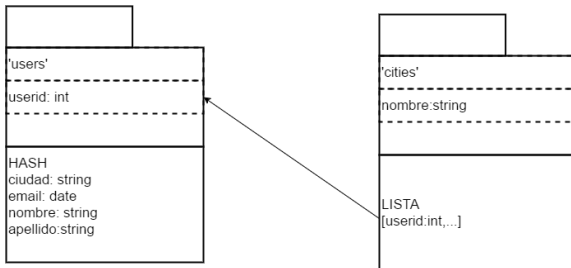
Notación Key-Value Ejemplo referencias



Notación Key-Value Ejemplo referencias

Users	
UserID	Info
7734	City:San Francisco, email:alef@gmail.com
4667	City:New York, email:samuel@yahoo.com
6578	City:Seattle, email:knovoselic@gmail.com
1263	City:San Francisco, email:eray@yahoo.com

Cities	
City	UserIDs
San Francisco	7734, 1263, ...
New York	4667, ...
Seattle	6578, ...



Ejercicio

Indices

Diseñar la base de datos para un twitter

- Se pueden usar como tipos de datos: Json, Hash, Sets
 - **HSET** *usuarios* nombre Jhon apellido Doe
 - **HGET** *usuarios* nombre \Rightarrow *Jhon*
- Se puede usar una operación: **INCR key**. (Clave tipo INCR en el diagrama)
 - **INCR** *prox_id* \Rightarrow 10