FRAGMENTACIÓN REPLICACIÓN

Autora: Cecilia Ruz

AGENDA

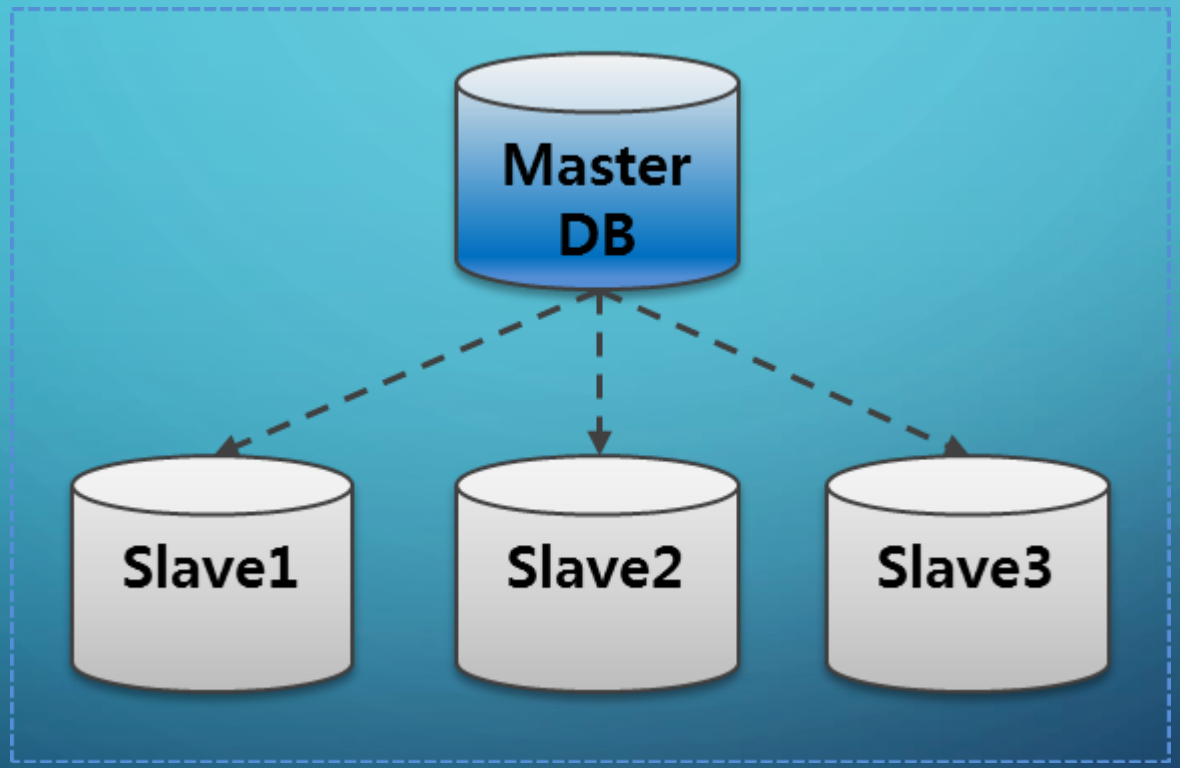
- INTRODUCCION
- REPLICACION
- FRAGMENTACION
- SHARDING
- METODOS DE SHARDING
- DESAFIOS

INTRODUCCIÓN

- El objetivo de esta clase es introducir el concepto de fragmentación y replicación y el porque de su necesidad. Mas adelante vamos a abordar los desafíos que plantea este tipo de organización.
- Supongamos que tengo una base de datos que esta llegando al límite de su capacidad, básicamente tengo 2 aproximaciones posibles
  - Aumentar el hardware del servidor en el que corre
  - Ver de que forma puedo “distribuir la carga de trabajo” entre diferentes servidores. Esto se puede conseguir
    - Creando más copias de los datos existentes (si el problema es de exceso de lectura)
    - Repartiendo los datos entre diferentes servidores (si tengo problemas de exceso de escritura)

REPLICACIÓN

- Hablamos de replicación cuando decido crear múltiples copias de los datos que tengo

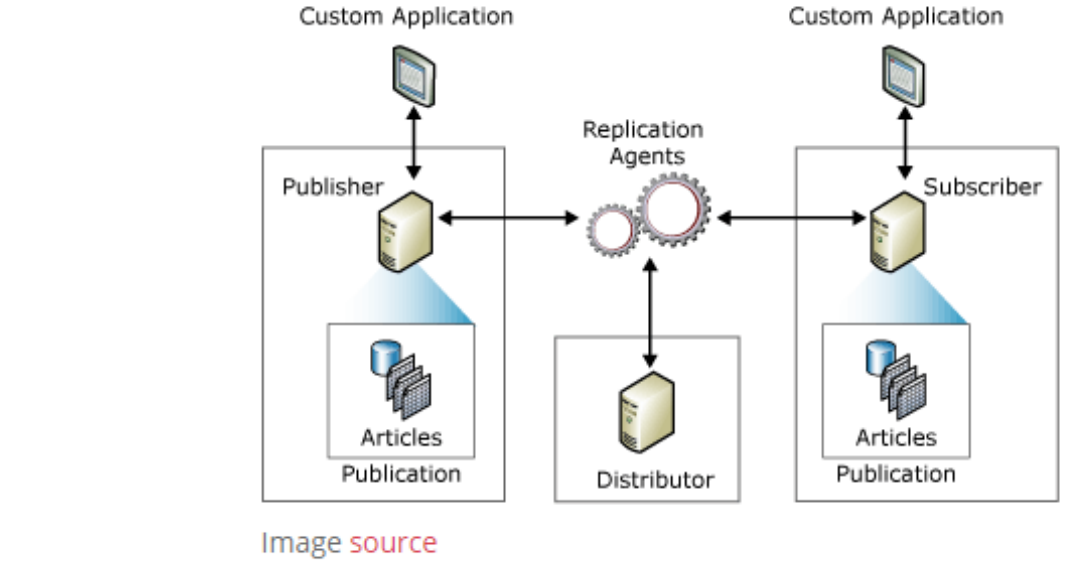


Fuente : http://solocodigoweb.com/blog/2017/05/22/el-poder-de-las-bases-de-datos-nosql/

MASTER - SLAVE

- En este mecanismo una única copia permite leer y escribir
- Las demás sólo permiten leer y reciben las actualizaciones de la copia principal
- En este tipo de replicación juegan un rol muy importante los logs
- En general existe un delay hasta que se actualizan las copias
- Si el nodo MASTER deja de estar disponible los slaves pueden elegir a otro nodo como su nuevo master
- Es posible realizar replicaciones que permitan leer y escribir en todas las copias, pero esas no corresponden al tipo MASTER - SLAVE

SQL SERVER



- Fuente : https://codingsight.com/sql-server-transactional-replication-internals/?utm_source=ssc&utm_medium=pubemail

FRAGMENTACIÓN

- Esta opción consiste en dividir los datos entre distintos servidores
- Cada fragmento puede estar en un nuevo servidor o en una parte de una arquitectura paralela en las bases de datos relacionales
- Tiene que haber un módulo encargado de administrar que fragmento tiene cada servidor
- Puede particionarse todo el esquema a partir del particionamiento de una tabla principal. Por ejemplo, a partir de los clientes se fragmentan las ventas, etc.
- La fragmentación puede ser vertical, horizontal o mixta

FRAGMENTACIÓN VERTICAL

branch_name	*customer_name*	*tuple_id*
Hillside	Lowman	1
Hillside	Camp	2
Valleyview	Camp	3
Valleyview	Kahn	4
Hillside	Kahn	5
Valleyview	Kahn	6
Valleyview	Green	7

$deposit_1 = \Pi_{branch\_name,\ customer\_name,\ tuple\_id}(employee\_info)$

account_number	*balance*	*tuple_id*
A-305	500	1
A-226	336	2
A-177	205	3
A-402	10000	4
A-155	62	5
A-408	1123	6
A-639	750	7

$deposit_2 = \Pi_{account\_number,\ balance,\ tuple\_id}(employee\_info)$

FRAGMENTACIÓN VERTICAL

- Otro tipo de fragmentación vertical consiste en separar tablas enteras.
- Por ejemplo
  - Un fragmento tiene todo lo que tiene que ver con las ventas
  - Otro lo de RRHH
  - Otro lo de compras
- No suele haber conjuntos completamente disjuntos. No hay más de 4 ó 5 particiones.

FRAGMENTACION HORIZONTAL

branch_name	*account_number*	*balance*
Hillside	A-305	500
Hillside	A-226	336
Hillside	A-155	62

$$account_1 = \sigma_{branch\_name="Hillside"}(account)$$

branch_name	*account_number*	*balance*
Valleyview	A-177	205
Valleyview	A-402	10000
Valleyview	A-408	1123
Valleyview	A-639	750

$$account_2 = \sigma_{branch\_name="Valleyview"}(account)$$

SHARDING

- Normalmente se llama sharding a la fragmentación horizontal
- El término se hizo popular de la mano de las NonSQL databases
- Está muy asociado a la escalabilidad horizontal, que permite agregar servidores de bajo costo para manejar mayor volumen de datos.

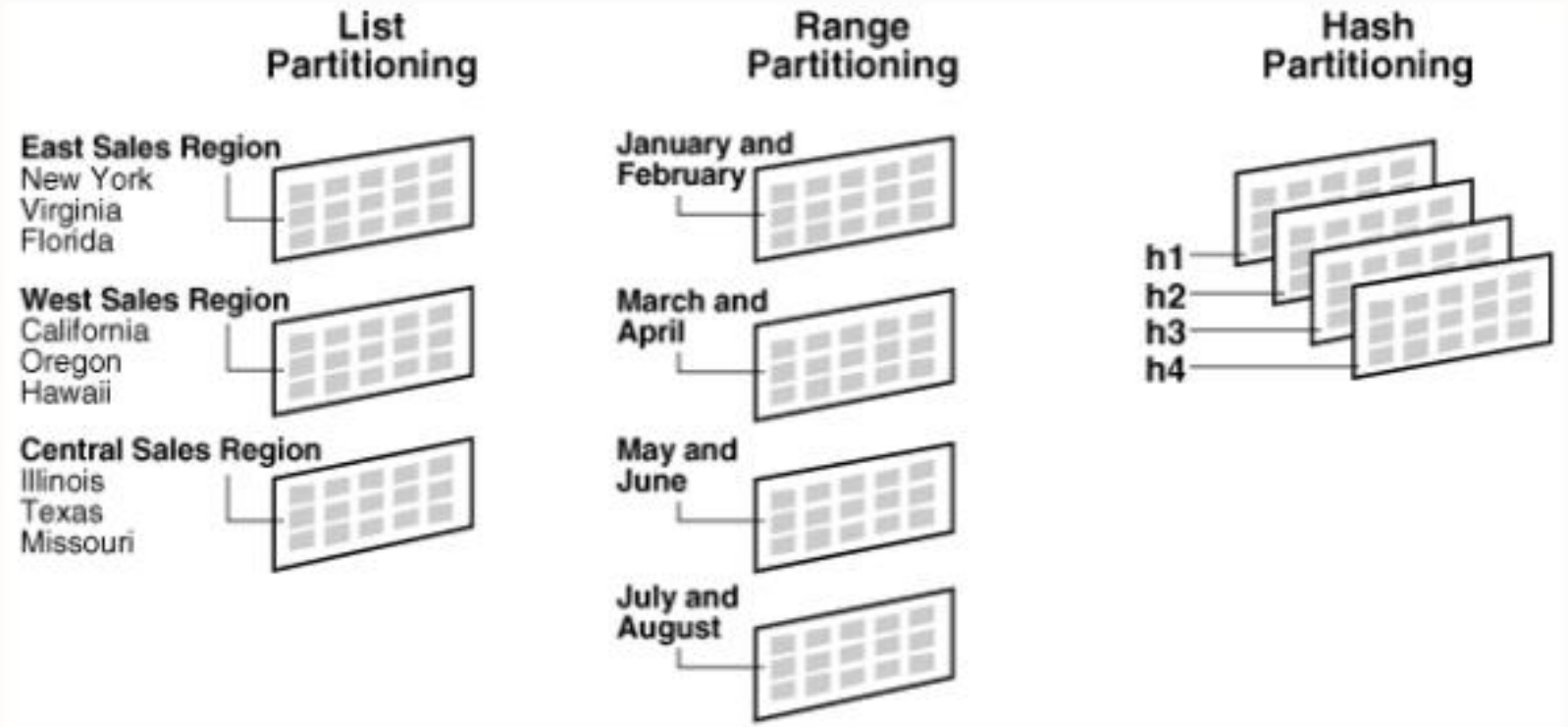
MECANISMOS DE SHARDING

- Siempre se trata de que los mecanismos produzcan “shards” balanceados
- Algunos tipos básicos son
  - Utilizando un rango de valores para un atributo de la tabla.
    - Por ejemplo: nro_cliente 1.1000, 1001….2000, etc.
  - Utilizando una lista de valores para un atributo de la tabla

Localidad	Partición
Bs.As, La Rioja	1
Cordoba, Santa fe	2
Mendoza, Chaco	3

  - Usando una función de hash sobre uno o más atributos de la tabla .

Figure 2-2 List, Range, and Hash Partitioning

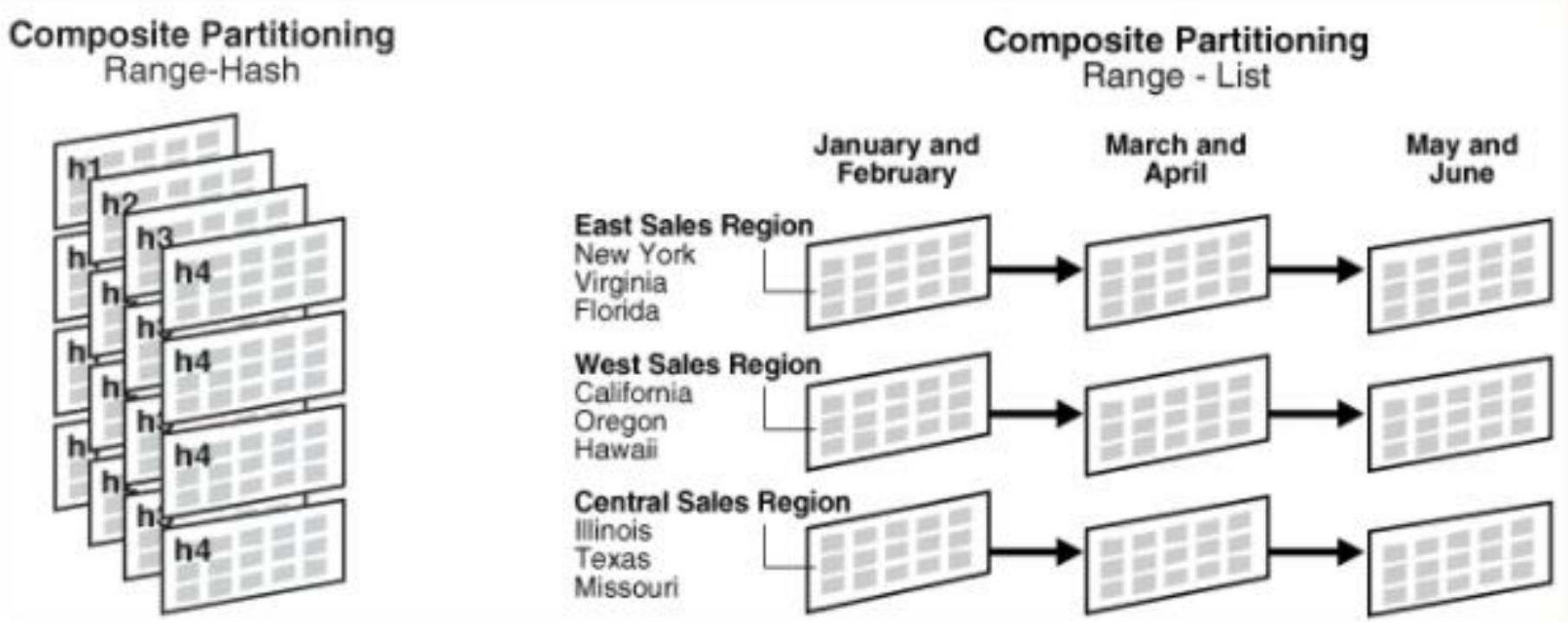


FUENTE : HTTPS://DOCS.ORACLE.COM/EN/DATABASE/ORACLE/ORACLE-DATABASE/21/VLDBG/PARTITION-CONCEPTS.HTML#GUID-D6CC12F9-81A5-4E89-872F-024C6161C0E8

MECANISMOS DE SHARDING

- SI fuera necesario puede efectuarse un “resharding” para mejorar el balanceo entre las particiones
- SI se efectúa un rebalanceo y uno está utilizando una función de hash se necesita guardar la versión anterior de la misma (o volver a distribuir todos los datos)
- En algunas bases de datos relacionales (por ejemplo Oracle, mysql), pueden aplicarse los métodos en cascada.

Figure 2-3 Composite Range—List Partitioning



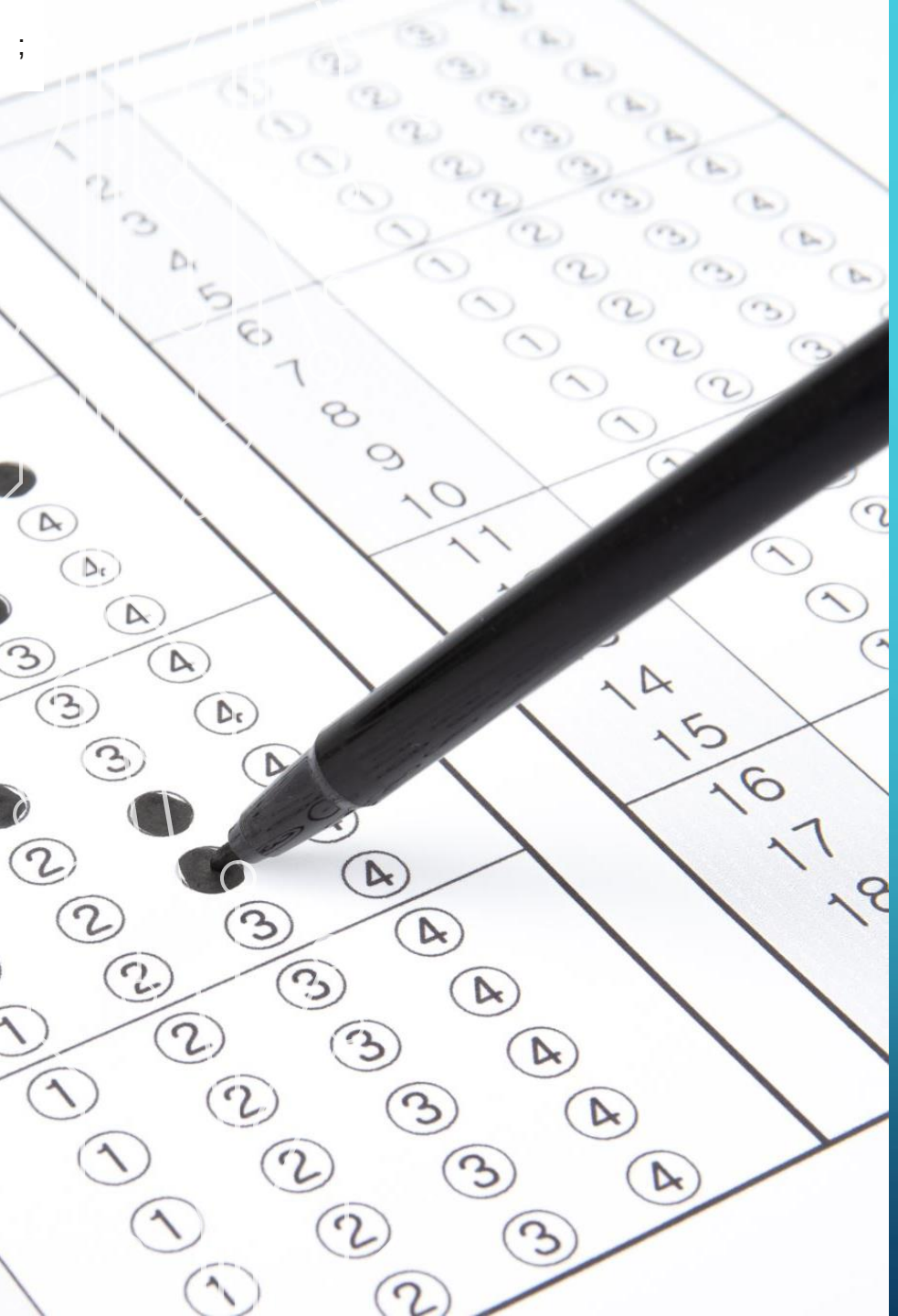
FUENTE : HTTPS://DOCS.ORACLE.COM/EN/DATABASE/ORACLE/ORACLE-DATABASE/21/VLDBG/PARTITION-CONCEPTS.HTML#GUID-D6CC12F9-81A5-4E89-872F-024C6161C0E8

EJEMPLO DE CREACIÓN DE PARTICIÓN

FUENTE: HTTPS://DOCS.ORACLE.COM/CD/E18283_01/SERVER.112/E16541/PART_ADMIN001.HTM#:~:TEXT=PARTITIONED%20TABLES%20PARTITION.-,CREATING%20COMPOSITE%20PARTITIONED%20TABLES,RANGE%20%7C%20LIST%20%7C%20HASH%5D%20CLAUSE

```
CREATE TABLE orders
( order_id NUMBER(12),
order_date TIMESTAMP WITH LOCAL TIME ZONE,
order_mode VARCHAR2(8),
customer_id NUMBER(6),
order_status NUMBER(2),
order_total NUMBER(8,2),
sales_rep_id NUMBER(6),
promotion_id NUMBER(6),
CONSTRAINT orders_pk PRIMARY KEY(order_id) )
PARTITION BY RANGE(order_date) (
PARTITION Q1_2005 VALUES LESS THAN (TO_DATE('01-APR-2005','DD-MON-YYYY')),
PARTITION Q2_2005 VALUES LESS THAN (TO_DATE('01-JUL-2005','DD-MON-YYYY')),
PARTITION Q3_2005 VALUES LESS THAN (TO_DATE('01-OCT-2005','DD-MON-YYYY')),
PARTITION Q4_2005 VALUES LESS THAN (TO_DATE('01-JAN-2006','DD-MON-YYYY'))
);
CREATE TABLE order_items
( order_id NUMBER(12) NOT NULL,
line_item_id NUMBER(3) NOT NULL,
product_id NUMBER(6) NOT NULL,
unit_price NUMBER(8,2),
quantity NUMBER(8),
CONSTRAINT order_items_fk FOREIGN KEY(order_id)
REFERENCES orders(order_id) )
PARTITION BY REFERENCE(order_items_fk);
```

;



EJEMPLO DE CONSULTA AL SYSTEM CATALOG

```
SELECT TABLE_NAME, PARTITIONING_TYPE, AUTOLIST, PARTITION_COUNT FROM USER_PART_TABLES WHERE TABLE_NAME ='orders'
```

SHARDING VS. REPLICACIÓN

- Estos métodos no son excluyentes entre si
- Puedo tener un conjunto de datos que además de estar fragmentado tenga varias réplicas de cada shard.

DESAFÍOS

- Resolución de consultas (con y sin join)
- Administración de transacciones
- Consistencia
- Iremos viendo cómo afecta el hecho de que los datos estén particionados o replicados a estos temas más adelante.

FRAGMENTACIÓN
REPLICACIÓN

MUCHAS GRACIAS