**TABLAS PARTICIONADAS**

**CONCEPTO**

Cuando tenemos tablas con un gran volumen de registros, una de las maneras más fáciles y eficaces de conseguir mejorar los tiempos de consultas, es particionar las tablas. Esto significa que, siguiendo un patrón que especifiquemos, Oracle va a dividir la tabla en tablas más pequeñas, desde el punto de vista físico, que desde el punto de vista lógico seguirá siendo una única tabla. Esto mejora considerablemente el tiempo de respuesta de la consulta, ya que cuando en esta filtra por el patrón especificado, Oracle consultará (físicamente) solo la partición correspondiente, y no se recorrerá la tabla completa.

¿Cuándo está recomendado usar tablas particionadas? Pues dependerá un poco del servidor sobre el que esté la base de datos, y del volumen de datos. Pero en principio, de forma genérica, podríamos decir que es recomendable a partir de 1 millón de registros.

***----------------------------------***

***-- 1. CREACION DE TBS***

***----------------------------------***

CREATE TABLESPACE TBS\_ASIA

DATAFILE 'C:\TEMP\DF\_COMPROBANTES\_ASIA.DBF'

SIZE 10M;

CREATE TABLESPACE TBS\_EUROPA

DATAFILE 'C:\TEMP\DF\_COMPROBANTES\_EUROPA.DBF'

SIZE 10M;

CREATE TABLESPACE TBS\_AL

DATAFILE 'C:\TEMP\DF\_COMPROBANTES\_AL.DBF'

SIZE 10M;

CREATE TABLESPACE TBS\_OTROS

DATAFILE 'C:\TEMP\DF\_COMPROBANTES\_OTROS.DBF'

SIZE 10M;

***----------------------------------***

***-- 2. TABLA PARTICIONADA POR LISTA***

***----------------------------------***

CREATE TABLE Ventas

(ID NUMBER(10),

ORIGEN VARCHAR2(20),

FECHA DATE default sysdate )

PARTITION BY LIST( ORIGEN)

(PARTITION ventas\_ASIA VALUES('ASIA') tablespace TBS\_ASIA,

PARTITION ventas\_EUROPA VALUES ('EUROPA') tablespace TBS\_EUROPA,

PARTITION ventas\_AL VALUES ('AL') tablespace TBS\_AL,

PARTITION ventas\_otros VALUES(DEFAULT) tablespace TBS\_OTROS );

***-------------------------------------------------***

***-- 3 INSERTANDO DATOS EN TABLAS PARTICIONADAS***

***-------------------------------------------------***

INSERT INTO VENTAS

SELECT LEVEL, 'ASIA', SYSDATE

FROM DUAL CONNECT BY LEVEL < 100000;

INSERT INTO VENTAS

SELECT LEVEL, 'EUROPA', SYSDATE

FROM DUAL CONNECT BY LEVEL < 100000;

***-------------------------------------------------***

***-- 4 QUERY DE TABLAS PARTICIONADAS***

***-------------------------------------------------***

SELECT \* FROM VENTAS;

SELECT \* FROM VENTAS PARTITION ( ventas\_ASIA );

SELECT \* FROM VENTAS PARTITION (ventas\_EUROPA);

***----------------------------------***

***-- 5 CONSULTANDO TABLAS PARTICIONADAS***

***----------------------------------***

SELECT TABLE\_NAME, PARTITION\_NAME, HIGH\_VALUE

FROM USER\_TAB\_PARTITIONS

WHERE TABLE\_NAME = 'VENTAS' ;

***----------------------------------***

***-- 6 COMPARANDO PLANES DE EJECUCION***

***-- ENTRE TABLA PARTICIONADA Y REGULAR***

***----------------------------------***

SELECT COUNT(\*) FROM VENTAS;

SELECT COUNT(\*) FROM VENTASREGULAR;

***---- PLAN DE EJECUCION TABLA REGULAR***

EXPLAIN PLAN

FOR

SELECT \* FROM VENTASREGULAR

WHERE VENTAS\_ORIGEN = 'ASIA';

SELECT \* FROM TABLE( DBMS\_XPLAN.DISPLAY )

***---- PLAN DE EJECUCION TABLA PARTICIONADA***

EXPLAIN PLAN

FOR

SELECT \* FROM VENTAS PARTITION ( VENTAS\_ASIA )

WHERE VENTAS\_ORIGEN = 'ASIA';

SELECT \* FROM TABLE( DBMS\_XPLAN.DISPLAY )

*VALORES COMPARATIVOS OBTENIDOS:*

Carga de 4 millones de registros en cada tabla.

Comparativo de Planes de Ejecución entre tablas Particionada y regular.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *TIPO* | *ROWS* | *BYTES* | *%CPU* | *TIME* |
| *T. REGULAR* | *2215K* | *71M* | *3890* | *00:00:47* |
| *T.PARTICIONADA* | *2244K* | *72M* | *1800* | *00:00:21* |

Conclusión:

* Las tablas particionadas consumen menos recursos en el uso de CPU y Tiempo de procesamiento ante una consulta.
* Se recomienda utilizar las estructuras de Tablas Particionadas para aquellas tablas que almacenan más de 1 millón de registros.

***------------------------------------------------***

***-- 7. CREACION DE TABLA PARTICIONADA POR RANGO***

***------------------------------------------------***

CREATE TABLE COMPROBANTES

( IDCOMPROBANTE NUMBER(15),

FECHA DATE DEFAULT SYSDATE ,

MONTO NUMBER(15,4) DEFAULT 0,

REGION VARCHAR(20) NOT NULL,

ESTADO INTEGER DEFAULT 1 )

PARTITION BY RANGE ( FECHA)

( PARTITION COMPROBANTES\_2016 VALUES

LESS THAN ( TO\_DATE( '2016-12-31 23:59:00', 'YYYY-MM-DD HH24:MI:SS' ))

TABLESPACE TBS\_COMPROBANTES\_2016,

PARTITION COMPROBANTES\_2017 VALUES

LESS THAN ( TO\_DATE( '2017-12-31 23:59:00', 'YYYY-MM-DD HH24:MI:SS' ))

TABLESPACE TBS\_COMPROBANTES\_2017,

PARTITION COMPROBANTES\_2018 VALUES

LESS THAN ( TO\_DATE( '2018-12-31 23:59:00', 'YYYY-MM-DD HH24:MI:SS' ))

TABLESPACE TBS\_COMPROBANTES\_2018 );

***----------------------------------***

***-- 5. INSERTANDO DATOS***

***----------------------------------***

INSERT INTO COMPROBANTES

SELECT LEVEL, SYSDATE, 350, 'AMERICA' , 1

FROM DUAL CONNECT BY LEVEL < 1000000;

INSERT INTO COMPROBANTES

SELECT LEVEL, TO\_DATE( '2017-2-5 23:59:00', 'YYYY-MM-DD HH24:MI:SS'), 350, 'AMERICA' , 1

FROM DUAL CONNECT BY LEVEL < 1000000;

INSERT INTO COMPROBANTES

SELECT LEVEL, TO\_DATE( '2018-2-5 23:59:00', 'YYYY-MM-DD HH24:MI:SS'), 350, 'AMERICA' , 1

FROM DUAL CONNECT BY LEVEL < 2000000;

***----------------------------------***

***-- 6. QUERY DE TABLAS PARTICIONADAS***

***----------------------------------***

SELECT \* FROM COMPROBANTES PARTITION (COMPROBANTES\_2016 );

***----------------------------------***

***-- 7. AGREGANDO UNA NUEVA PARTICION***

***----------------------------------***

CREATE TABLESPACE TBS\_COMPROBANTES\_2019

DATAFILE 'C:\TEMP\DF\_COMPROBANTES\_2019.DBF'

SIZE 100M;

ALTER TABLE COMPROBANTES ADD PARTITION COMPROBANTES\_2019

VALUES LESS THAN ( TO\_DATE( '2017-12-31 23:59:00', 'YYYY-MM-DD HH24:MI:SS' )) TABLESPACE TBS\_COMPROBANTES\_2019 ;

SELECT TABLE\_NAME, PARTITION\_NAME, HIGH\_VALUE

FROM USER\_TAB\_PARTITIONS

WHERE TABLE\_NAME = 'COMPROBANTES'

INSERT INTO COMPROBANTES

SELECT LEVEL, TO\_DATE( '2017-2-5 23:59:00', 'YYYY-MM-DD HH24:MI:SS'), 350, 'AMERICA' , 1

FROM DUAL CONNECT BY LEVEL < 2;