**Flashback Data Archive**

**Introduction**

Flashback Data Archive is a new feature in Oracle Database 11g that automatically track and maintain changes to data in a user/application transparent manner. This provides robust and secure mechanism for tracking and querying historical data for any database table, without any performance impact of traditional approaches.

Flashback Data Archive provides a centralized and policy-based management interface to automate ongoing administrative tasks. With Flashback Data Archive, you can easily group tables and set a common retention policy for each group. When a new table is added to a flashback data archive, it automatically inherits the retention parameter from that archive.

**Architecture**

Flashback Data Archive works with undo data as do most other Flashback features. However, the historical data archive is not limited by the size of the undo tablespace.

Undo Data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  | Buffer Cache |
|  |  |  |  |  |
|  | D | M | L |  |
|  |  |  |  |  |
|  |  |  |  |  |

FBDA Background Process

Historical Data

**Benefits**

Enabling Flashback Data Archive on one or more tables can be done instantaneously with no application changes.

Accurate and complete history tracking

Viewing history is also complicated: querying and reporting on the data is hard, and viewing data across multiple history tables is even more difficult.

High implementation costs, high maintenance costs, performance limitations, and complexity.

Direct updates to the data at the database level could introduce data integrity issues with the history, and could invalidate the entire set of historical data.

Application upgrades, changing retention requirements, Adding/dropping a new column, adding/dropping the constraints,

Error recovery: restore records that were erroneously removed or updated

Historical reporting: analyse product changes over time

Retention policy enforcement: automatically purge history older than 5 years

Flashback Data Archive employs a lightweight mechanism to mark DML operations on FDA enabled tables.

Storage optimization –Flashback Data Archive compresses and partitions the internal history tables to optimize storage and performance. Both compression and partitioning in Flashback Data Archive are managed automatically and require no special administration.

Providing self-service error correction for an application, thereby enabling users to undo and correct their errors??

You can run a daily report that shows the change in data from yesterday. You can compare individual rows of table data or find intersections or unions of sets of rows??

Simplifying application design by removing the need to store some kinds of temporal data?? retrieve past data directly from the same table?

**Step-by-Step Approach**

Enabling Flashback Data Archive is just a three step process.

1. Either create a new tablespace or use an existing one.

2. Create a flashback data archive:

create flashback archive AMS\_FLASH\_ARCHIVE tablespace AMS\_ARCH retention 1 year;

3. Enable Flashback Data Archive on the desired tables.

ALTER TABLE archive\_test FLASHBACK ARCHIVE AMS\_FLASH\_ARCHIVE;

**Accessing History Data**

select \* from archive\_test as of timestamp to\_timestamp('30032015 15:54:59','ddmmyyyy hh24:mi:ss');

select \* from archive\_test as of timestamp to\_timestamp('30032015 15:55:31','ddmmyyyy hh24:mi:ss');

select \* from archive\_test VERSIONS BETWEEN timestamp to\_timestamp('30032015 15:54:59','ddmmyyyy hh24:mi:ss')

and to\_timestamp('30032015 15:55:31','ddmmyyyy hh24:mi:ss')

where col1='C1';

**Risks, Issues, Concerns**

Is history data storage optimized? (Partitioned, compressed, etc)

Performance impact as volume of history data increases.

How are we ensuring access to history only through read-only methods?

How are we ensuring referential integrity constraints, cross table history data?

Effect of Add, Drop, Rename, Modify Column?

Effect of Rename, Truncate, Drop Table?

How to access the history data?

How to rollback an undesired change using history data?

How the retention policy is implemented and how to change the retention policy?

How to adopt the changing business needs and new regulations?

What if you find a DDL that can corrupt the history data?

Examples

Suppose that you discover at 12:30 PM that the row for employee Chung was deleted from the employees table, and you know that at 9:30AM the data for Chung was correctly stored in the database.

SELECT \* FROM employees AS OF TIMESTAMP TO\_TIMESTAMP('2004-04-04 09:30:00', 'YYYY-MM-DD HH:MI:SS')

WHERE last\_name = 'Chung';

Restoring a Lost Row After Oracle Flashback Query

INSERT INTO employees (

SELECT \* FROM employees

AS OF TIMESTAMP

TO\_TIMESTAMP('2004-04-04 09:30:00', 'YYYY-MM-DD HH:MI:SS')

WHERE last\_name = 'Chung'

);

If a table is a Flashback Data Archive and you specify a time for it that is earlier than its creation time, the query returns zero rows for that table, rather than causing an error.

You can create a view that refers to past data by using the AS OF clause in the SELECT statement that defines the view.

CREATE VIEW hour\_ago AS

SELECT \* FROM employees

AS OF TIMESTAMP (SYSTIMESTAMP - INTERVAL '60' MINUTE);

This query reinserts into table employees the rows that existed an hour ago:

INSERT INTO employees

(SELECT \* FROM employees

AS OF TIMESTAMP (SYSTIMESTAMP - INTERVAL '60' MINUTE)

MINUS SELECT \* FROM employees;

Oracle Flashback Version Query Row Data Pseudocolumns

VERSIONS\_STARTSCN : Starting System Change Number (SCN) or TIMESTAMP when the row version was created.

VERSIONS\_STARTTIME

VERSIONS\_ENDSCN : SCN or TIMESTAMP when the row version expired.

VERSIONS\_ENDTIME

VERSIONS\_OPERATION : Operation performed by the transaction: I for insertion, D for deletion, or U for update. The version is that of the row that was inserted, deleted, or updated; that is, the row after an INSERT operation, the row before a DELETE operation, or the row affected by an UPDATE operation. For user updates of an index key, Oracle Flashback Version Query might treat an UPDATE operation as two operations, DELETE plus INSERT, represented as two version rows with a D followed by an I VERSIONS\_OPERATION.

SELECT versions\_startscn, versions\_starttime,

versions\_endscn, versions\_endtime,

versions\_xid, versions\_operation,

last\_name, salary

FROM employees

VERSIONS BETWEEN TIMESTAMP

TO\_TIMESTAMP('2008-12-18 14:00:00', 'YYYY-MM-DD HH24:MI:SS')

AND TO\_TIMESTAMP('2008-12-18 17:00:00', 'YYYY-MM-DD HH24:MI:SS')

WHERE first\_name = 'John';

You can use VERSIONS\_XID with Oracle Flashback Transaction Query to locate this transaction's metadata, including the SQL required to undo the row change and the user responsible for the change.

SELECT xid, operation, start\_scn, commit\_scn, logon\_user, undo\_sql

FROM flashback\_transaction\_query

WHERE xid = HEXTORAW('000200030000002D');

This statement uses Oracle Flashback Version Query as a subquery to associate each row version with the LOGON\_USER responsible for the row data change:

SELECT xid, logon\_user

FROM flashback\_transaction\_query

WHERE xid IN (

SELECT versions\_xid FROM employees VERSIONS BETWEEN TIMESTAMP

TO\_TIMESTAMP('2003-07-18 14:00:00', 'YYYY-MM-DD HH24:MI:SS') AND

TO\_TIMESTAMP('2003-07-18 17:00:00', 'YYYY-MM-DD HH24:MI:SS')

);

**Using DBMS\_FLASHBACK Package**

The DBMS\_FLASHBACK package provides the same functionality as Oracle Flashback Query, but Oracle Flashback Query is sometimes more convenient.

The DBMS\_FLASHBACK package acts as a time machine: you can turn back the clock, perform normal queries as if you were at that earlier time, and then return to the present.

1. Specify a past time by invoking either DBMS\_FLASHBACK.ENABLE\_AT\_TIME or DBMS\_FLASHBACK.ENABLE\_AT\_SYSTEM\_CHANGE\_NUMBER.

2. Perform regular queries (that is, queries without special flashback-feature syntax such as AS OF). Do not perform DDL or DML operations.

3. Return to the present by invoking DBMS\_FLASHBACK.DISABLE.

You must invoke DBMS\_FLASHBACK.DISABLE before invoking DBMS\_FLASHBACK.ENABLE\_AT\_TIME or DBMS\_FLASHBACK.ENABLE\_AT\_SYSTEM\_CHANGE\_NUMBER again. You cannot nest enable/disable pairs.

**Using Flashback Data Archive (Oracle Total Recall)**

You can enable flashback archiving for a table if all of these conditions are true:

• You have the FLASHBACK ARCHIVE object privilege on the Flashback Data Archive to use for that table.

• The table is neither nested, clustered, temporary, remote, or external.

• The table contains neither LONG nor nested columns.

• The table does not use any of these Flashback Data Archive reserved words as column names: STARTSCN, ENDSCN, RID, XID, OP, OPERATION

After flashback archiving is enabled for a table, you can disable it only if you either have the FLASHBACK ARCHIVE ADMINISTER system privilege or you are logged on as SYSDBA.

Altering a Flashback Data Archive

•Create a default Flashback Data Archive named fla1 that uses up to 10 G of tablespace tbs1, whose data are retained for one year:

CREATE FLASHBACK ARCHIVE DEFAULT fla1 TABLESPACE tbs1 QUOTA 10G RETENTION 1 YEAR;

• Make Flashback Data Archive fla1 the default Flashback Data Archive:

ALTER FLASHBACK ARCHIVE fla1 SET DEFAULT;

• To Flashback Data Archive fla1, add up to 5 G of tablespace tbs3:

ALTER FLASHBACK ARCHIVE fla1 ADD TABLESPACE tbs3 QUOTA 5G;

• To Flashback Data Archive fla1, add as much of tablespace tbs4 as needed:

ALTER FLASHBACK ARCHIVE fla1 ADD TABLESPACE tbs4;

• Change the maximum space that Flashback Data Archive fla1 can use in tablespace tbs3 to 20 G:

ALTER FLASHBACK ARCHIVE fla1 MODIFY TABLESPACE tbs3 QUOTA 20G;

• Allow Flashback Data Archive fla1 to use as much of tablespace tbs1 as needed:

ALTER FLASHBACK ARCHIVE fla1 MODIFY TABLESPACE tbs1;

• Change the retention time for Flashback Data Archive fla1 to two years:

ALTER FLASHBACK ARCHIVE fla1 MODIFY RETENTION 2 YEAR;

• Remove tablespace tbs2 from Flashback Data Archive fla1:

ALTER FLASHBACK ARCHIVE fla1 REMOVE TABLESPACE tbs2;

(Tablespace tbs2 is not dropped.)

• Purge all historical data from Flashback Data Archive fla1:

ALTER FLASHBACK ARCHIVE fla1 PURGE ALL;

• Purge all historical data older than one day from Flashback Data Archive fla1:

ALTER FLASHBACK ARCHIVE fla1 PURGE BEFORE TIMESTAMP (SYSTIMESTAMP - INTERVAL '1' DAY);

• Purge all historical data older than SCN 728969 from Flashback Data Archive fla1:

ALTER FLASHBACK ARCHIVE fla1 PURGE BEFORE SCN 728969;

Drop a Flashback Data Archive with the DROP FLASHBACK ARCHIVE statement. Dropping a Flashback Data Archive deletes its historical data, but does not drop its tablespaces.

DROP FLASHBACK ARCHIVE fla1;

Enabling and Disabling Flashback Data Archive

By default, flashback archiving is disabled for any table. You can enable flashback archiving for a table if you have the FLASHBACK ARCHIVE object privilege on the Flashback Data Archive to use for that table.

To enable flashback archiving for a table, include the FLASHBACK ARCHIVE clause in either the CREATE TABLE or ALTER TABLE statement. In the FLASHBACK ARCHIVE clause, you can specify the Flashback Data Archive where the historical data for the table are stored. The default is the default Flashback Data Archive for the system.

Examples

• Create table employee and store the historical data in the default Flashback Data Archive:

CREATE TABLE employee (EMPNO NUMBER(4) NOT NULL, ENAME VARCHAR2(10),

JOB VARCHAR2(9), MGR NUMBER(4)) FLASHBACK ARCHIVE;

• Create table employee and store the historical data in the Flashback Data Archive fla1:

CREATE TABLE employee (EMPNO NUMBER(4) NOT NULL, ENAME VARCHAR2(10),

JOB VARCHAR2(9), MGR NUMBER(4)) FLASHBACK ARCHIVE fla1;

• Enable flashback archiving for the table employee and store the historical data in the default Flashback Data Archive:

ALTER TABLE employee FLASHBACK ARCHIVE;

• Enable flashback archiving for the table employee and store the historical data in the Flashback Data Archive fla1:

ALTER TABLE employee FLASHBACK ARCHIVE fla1;

• Disable flashback archiving for the table employee:

ALTER TABLE employee NO FLASHBACK ARCHIVE;

**DL Statements on Tables Enabled for Flashback Data Archive**

Flashback Data Archive supports only these DDL statements:

• ALTER TABLE statement that does any of the following:

• Adds, drops, renames, or modifies a column

• Adds, drops, or renames a constraint

• Drops or truncates a partition or subpartition operation

• TRUNCATE TABLE statement

• RENAME statement that renames a table

Using Flashback Data Archive to Recover Data

DELETE EMPLOYEE WHERE MANAGER = 'LISA JOHNSON';

INSERT INTO EMPLOYEE

SELECT \* FROM EMPLOYEE

AS OF TIMESTAMP (SYSTIMESTAMP - INTERVAL '2' DAY)

WHERE MANAGER = 'LISA JOHNSON';

SELECT \* FROM employees@some\_remote\_host AS OF

TIMESTAMP (SYSTIMESTAMP - INTERVAL '60' MINUTE);

To query past data at a precise time, use an SCN. If you use a time stamp, the actual time queried might be up to 3 seconds earlier than the time you specify. Oracle Database uses SCNs internally and maps them to time stamps at a granularity of 3 seconds.

For example, suppose that the SCN values 1000 and 1005 are mapped to the time stamps 8:41 AM and 8:46 AM, respectively. A query for a time between 8:41:00 and 8:45:59 AM is mapped to SCN 1000; an Oracle Flashback Query for 8:46 AM is mapped to SCN 1005. Therefore, if you specify a time that is slightly after a DDL operation (such as a table creation) Oracle Database might use an SCN that is just before the DDL operation, causing error ORA-01466.

|  |  |  |
| --- | --- | --- |
|  | FDA | MPF Snapping Process |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |