

Using Flashback Technology I

8

Copyright © 2017, BCI LTD.. All rights reserved.



Objectives

Flashback
> - Overview
- Query
- Table
- Transaction

After completing this lesson, you should be able to:

- Describe Flashback technology
- Perform Flashback Query
- Use Flashback Version Query
- Enable row movement on a table
- Perform Flashback Table operations
- Use Flashback Transaction Query
- Use Flashback Transaction



Flashback Technology

Object Level	Scenario Examples	Flashback Technology	Depends On	Affects Data
Database	Truncate table; Undesired multitable changes made	Database	Flashback logs	TRUE
Table	Drop table	Drop	Recycle bin	TRUE
	Update with the wrong WHERE clause	Table	Undo data	TRUE
	Compare current data with data from the past	Query	Undo data	FALSE
	Compare versions of a row	Version	Undo data	FALSE
	Keep historical transaction data	Data Archive	Undo data	TRUE
Transaction	Investigate and back out suspect transactions	Transaction	Undo/redo from Archive logs	TRUE

8 - 3

Copyright © 2017, BCI LTD.. All rights reserved.

Flashback Technology

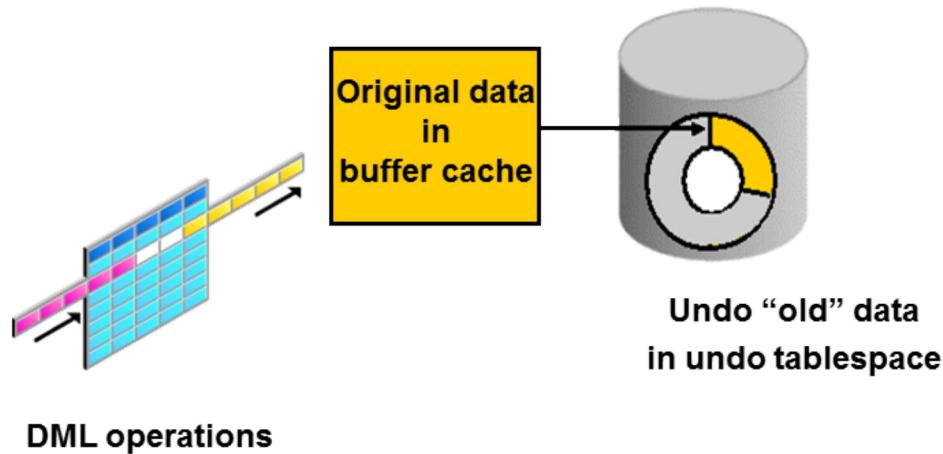
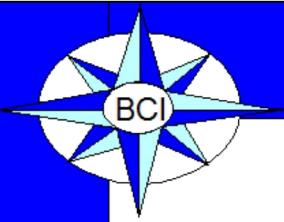
You can use Flashback technology when a logical corruption occurs in the Oracle database and you need to recover data quickly and easily. As with human errors, it is difficult to identify the objects and rows that are affected by an erroneous transaction. With Flashback technology, you can diagnose how errors are introduced into the database, and then repair the damage. You can view the transactions that have contributed to specific row modifications, view the entire set of versions of a given row during a specific time period, or just view data as it appeared at a specific time in the past. The table in the slide shows typical uses of Flashback technology. Flashback Database depends on the flashback logs to perform flashback. Flashback Drop uses the recycle bin. All other techniques use undo data.

Not all flashback features modify the database. Some are simply methods to query other versions of data; these are tools to investigate a problem and aid in recovery. The results of flashback queries help you do one of two things:

- Determine the type of database-modifying flashback operation to perform to fix the problem.
- Feed the result set of these queries into an INSERT, UPDATE, or DELETE statement that enables you to easily repair the erroneous data.

Flashback Data Archive enables you to use the preceding logical flashback features to access data from far back in the past.

Transactions and Undo



8 - 4

Copyright © 2017, BCI LTD.. All rights reserved.

Transactions and Undo

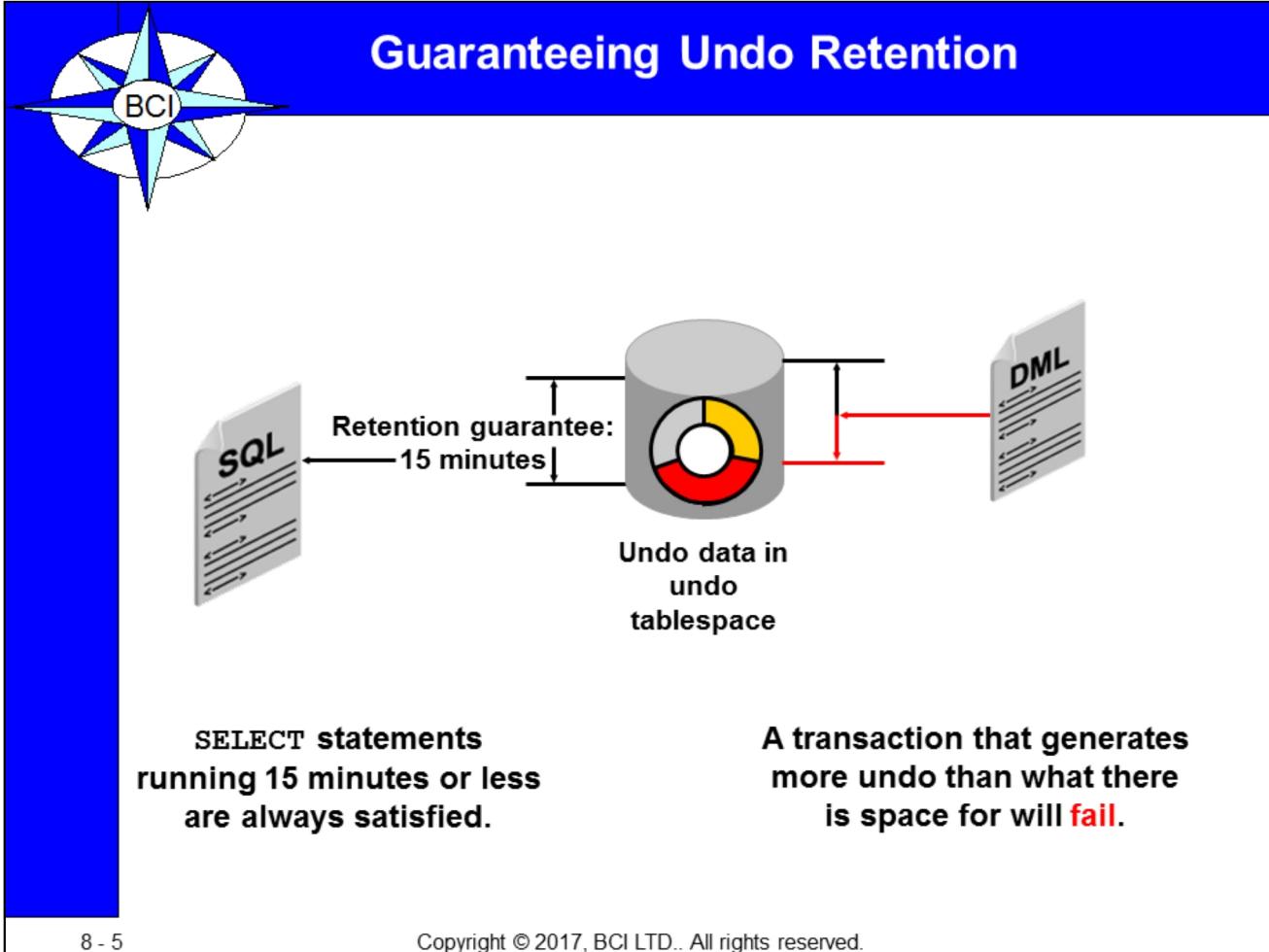
When a transaction starts, it is assigned to an undo segment. Throughout the life of the transaction, when data is changed, the original “old” values are copied into the undo segment. You can see which transactions are assigned to which undo segments by checking the V\$TRANSACTION view.

Undo segments are specialized segments that are automatically created by the instance as needed to support transactions. Like all segments, undo segments are made up of extents, which, in turn, consist of data blocks. Undo segments automatically grow and shrink as needed, acting as a circular storage buffer for their assigned transactions.

When transactions fill the blocks in their current undo segment extent, they are assigned another block in the same extent. If no free blocks remain in that extent, the transaction acquires a block from the next extent in the segment. If all extents are in use, the transaction either wraps around back into the first extent or requests that a new extent be allocated to the undo segment.

The diagram in the slide shows on the left a table icon with original data arriving from a DML operation. The original data is kept in the buffer cache (if not aged out) and then written to the undo tablespace (shown in circular form on the right).

Note: Parallel DML operations can actually cause a transaction to use more than one undo segment. To learn more about parallel DML execution, see the *Oracle Database Administrator’s Guide*.



Guaranteeing Undo Retention

The default undo behavior is to overwrite committed transactions that have not yet expired rather than to allow an active transaction to fail because of lack of undo space. In case of conflict, transactions have precedence over queries.

This behavior can be changed by guaranteeing retention. With guaranteed retention, undo retention settings are enforced even if they cause transactions to fail. (So in case of conflict, queries have precedence over transactions.)

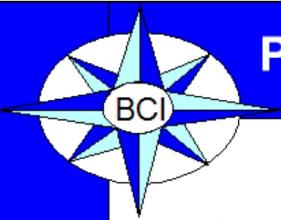
RETENTION GUARANTEE is a tablespace attribute rather than an initialization parameter. This attribute can be changed using either SQL command-line statements or Enterprise Manager. The syntax to change an undo tablespace to guarantee retention is:

```
SQL> ALTER TABLESPACE undotbs1 RETENTION GUARANTEE;
```

To return a guaranteed undo tablespace to its normal setting, use the following command:

```
SQL> ALTER TABLESPACE undotbs1 RETENTION NOGUARANTEE;
```

You can set Undo Retention Guarantee in Enterprise Manager. Navigate to the Automatic Undo Management page. Click the current setting for Retention Guarantee (General/Undo Retention Settings) to modify it.



Preparing Your Database for Flashback

- Creating an undo tablespace
- Enabling Automatic Undo Management
- Specifying versus guaranteeing undo retention

Default database initialization parameters:

- UNDO_MANAGEMENT='AUTO'
- UNDO_TABLESPACE='UNDOTBS1'
- UNDO_RETENTION=900

Preparing Your Database for Flashback

To enable flashback features for an application, you must perform these tasks:

- Create an undo tablespace with enough space to keep the required data for flashback operations. The more often users update the data, the more space is required. The database administrator usually calculates the space requirement. If you are uncertain about your space requirements, you can start with an automatically extensible undo tablespace, observe it through one business cycle (for example, 1 or 2 days), collect undo block information with the V\$UNDO_STAT view, calculate your space requirements, and use them to create an appropriately sized fixed undo tablespace. (The calculation formula is in the *Oracle Database Administrator's Guide*.)
- By default, Automatic Undo Management is enabled. If needed, enable Automatic Undo Management, as explained in the *Oracle Database Administrator's Guide*.
- For a fixed-size undo tablespace, the Oracle database automatically tunes the system to give the undo tablespace the best possible undo retention.
- For an automatically extensible undo tablespace (default), the Oracle database retains undo data to satisfy at a minimum, the retention periods needed by the longest-running query and the threshold of undo retention, specified by the UNDO_RETENTION parameter.

Preparing Your Database for Flashback (continued)

You can query V\$UNDOSTAT.TUNED_UNDORETENTION to determine the amount of time for which undo is retained for the current undo tablespace. Setting the UNDO_RETENTION parameter does not guarantee, that unexpired undo data is not overwritten. If the system needs more space, the Oracle database can overwrite unexpired undo with more recently generated undo data.

- Specify the RETENTION GUARANTEE clause for the undo tablespace to ensure that unexpired undo data is not discarded.
- Grant flashback privileges to users, roles, or applications that need to use flashback features.

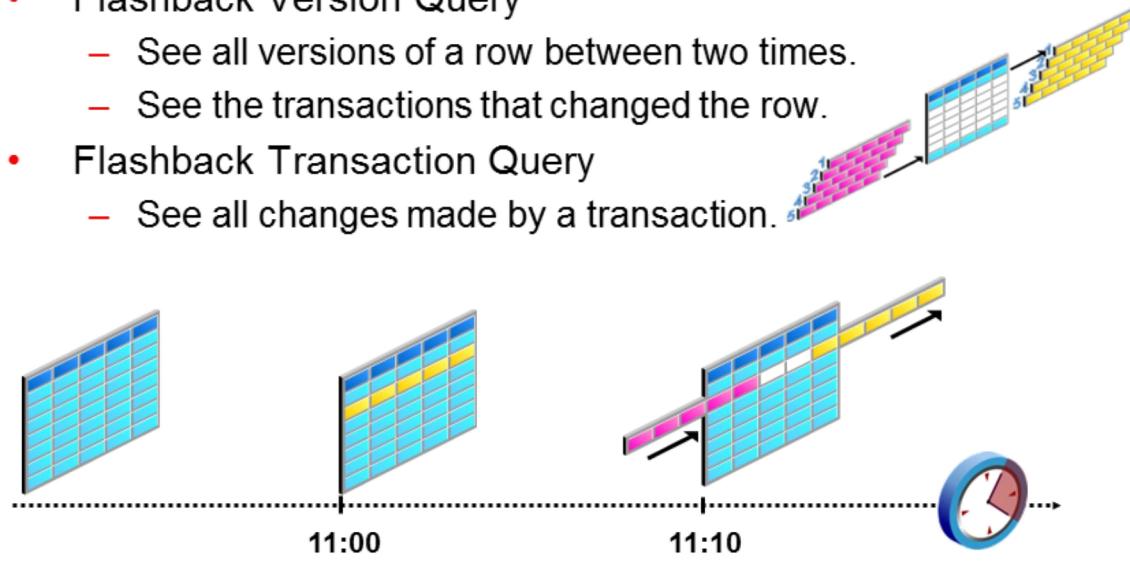
To satisfy long retention requirements, create a Flashback Data Archive.



Using Flashback Technology to Query Data

Flashback
- Overview
-> - **Query**
- Table
- Transaction

- Flashback Query
 - Query all data at a specified point in time.
- Flashback Version Query
 - See all versions of a row between two times.
 - See the transactions that changed the row.
- Flashback Transaction Query
 - See all changes made by a transaction.



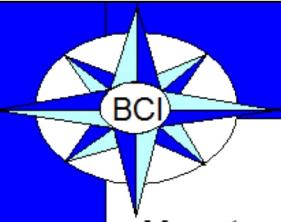
8 - 8

Copyright © 2017, BCI LTD.. All rights reserved.

Using Flashback Technology to Query Data

Flashback technology provides the capability to query past versions of schema objects, query historical data, and perform change analysis. Every transaction logically generates a new version of the database. With Flashback technology, you can navigate through these versions to find an error and its cause:

- **Flashback Query:** Query all data as it existed at a specific point in time.
- **Flashback Version Query:** See all versions of rows between two times and the transactions that changed the row.
- **Flashback Transaction Query:** See all changes made by a transaction and, if needed, roll back a transaction with “undo” SQL commands.



Flashback Query

Use to query all data at a specified point in time.



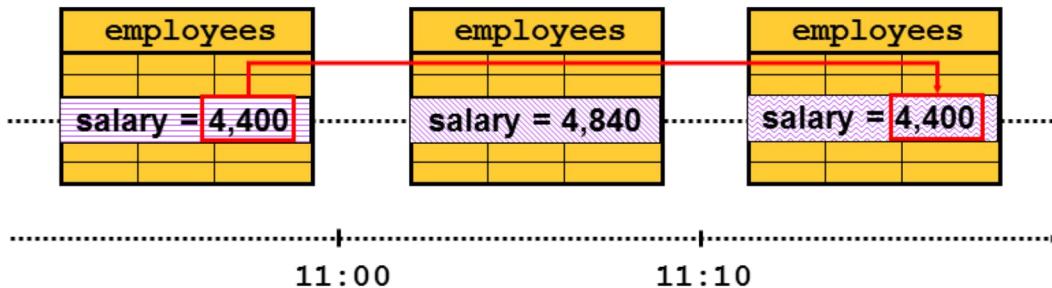
```
SELECT employee_id, salary FROM employees  
AS OF TIMESTAMP <T1>  
WHERE employee_id = 200
```

Flashback Query

With the Flashback Query feature, you can perform queries as of a certain time. By using the AS OF clause of the SELECT statement, you can specify the time stamp for which to view the data. This is useful for analyzing a data discrepancy.

Note: TIMESTAMP and SCN are valid options for the AS OF clause.

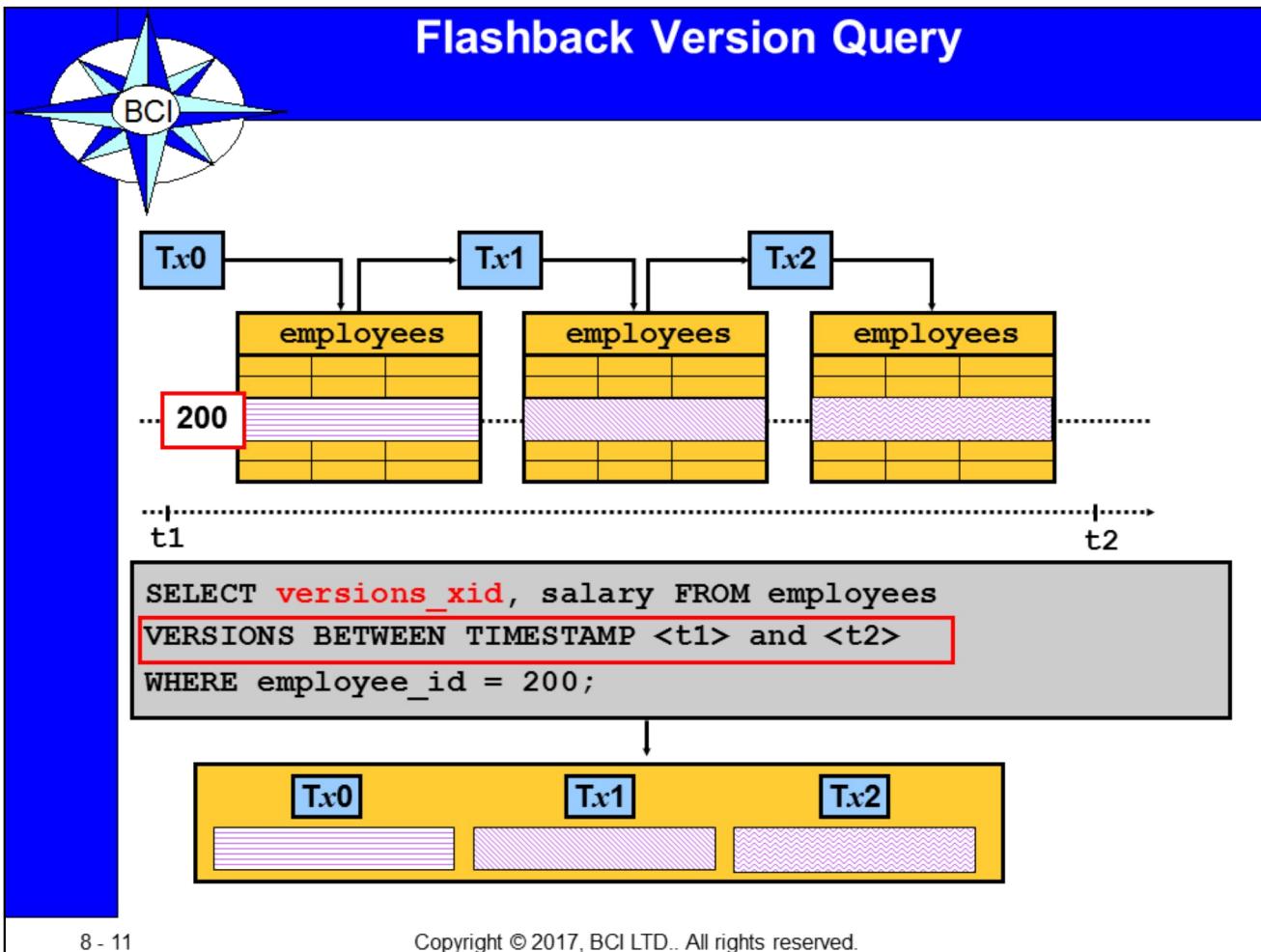
Flashback Query: Example



```
UPDATE employees
SET salary =
  (SELECT salary FROM employees
   AS OF TIMESTAMP TO_TIMESTAMP
   ('2016-05-04 11:00:00', 'yyyy-mm-dd hh24:mi:ss')
   WHERE employee_id = 200)
WHERE employee_id = 200
```

Flashback Query: Example

If a raise has been erroneously given to a particular employee recently, you can update the salary again, assigning the salary provided by a subquery that returns the flashed-back value.



Flashback Version Query

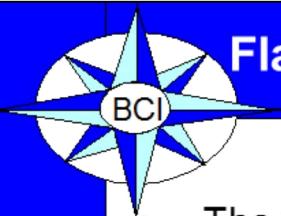
With Flashback Query, you can perform queries on the database as of a certain time span or range of user-specified system change numbers (SCNs). The Flashback Version Query feature enables you to use the VERSIONS clause to retrieve all the versions of the rows that exist between two points in time or two SCNs.

The rows returned by Flashback Version Query represent a history of changes for the rows across transactions. Flashback Version Query retrieves only committed occurrences of the rows.

Uncommitted row versions within a transaction are not shown. The rows returned also include deleted and subsequently reinserted versions of the rows.

You can use Flashback Version Query to retrieve row history. It provides you with a way to audit the rows of a table and retrieve information about the transactions that affected the rows. You can then use the returned transaction identifier to perform transaction mining by using LogMiner or to perform a Flashback Transaction Query, as described later in this lesson.

Note: VERSIONS_XID is a pseudocolumn that returns the transaction identifier of the corresponding version of a row.



Flashback Version Query: Considerations

- The VERSIONS clause cannot be used to query:
 - External tables
 - Temporary tables
 - Fixed tables
 - Views
- The VERSIONS clause cannot span DDL commands.
- Segment shrink operations are filtered out.

Flashback Version Query: Considerations

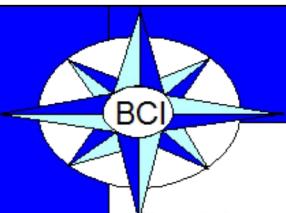
The VERSIONS clause cannot be used to query the following types of tables:

- External tables
- Temporary tables
- Fixed tables

You cannot use the VERSIONS clause to query a view. However, a view definition can use the VERSIONS clause.

The VERSIONS clause in a SELECT statement cannot produce versions of rows across the DDL statements that change the structure of the corresponding tables. This means that the query stops producing rows after it reaches a time in the past when the table structure was changed.

Certain maintenance operations, such as a segment shrink, may move table rows across blocks. In this case, the version query filters out such phantom versions because the row data remains the same.



Quiz

Flashback Query compares current data with data from the past. To do so, it uses both undo and redo data.

1. True
2. False

Answer: 2

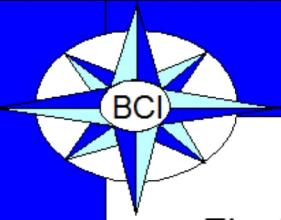


Quiz

Select the correct statement:

1. Flashback Version Query uses undo data and modifies data.
2. Flashback Version Query uses undo data and does not modify data.
3. Flashback Version Query uses both undo and redo data.

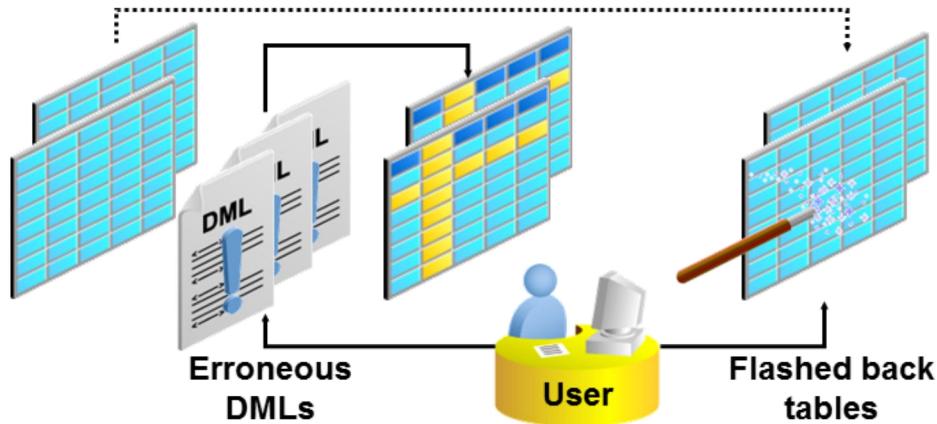
Answer: 2



Flashback Table: Overview

Flashback
- Overview
- Query
> - Table
- Transaction

- Flashback Table recovers tables to a specific point in time.
- Flashback Table is an in-place operation.
- The database stays online.



8 - 15

Copyright © 2017, BCI LTD.. All rights reserved.

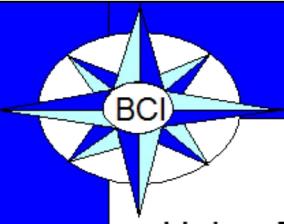
Flashback Table Overview

Using Flashback Table, you can recover a set of tables to a specific point in time without having to perform traditional point-in-time recovery operations.

A Flashback Table operation is done in-place, while the database is online, by rolling back only the changes that are made to the given tables and their dependent objects.

A Flashback Table statement is executed as a single transaction. All tables must be flashed back successfully, or the entire transaction is rolled back.

Note: You can use Flashback Versions Query and Flashback Transaction Query to determine the appropriate flashback time.



Flashback Table

- Using Flashback Table, you can recover a table or tables to a specific point in time without restoring a backup.
- Data is retrieved from the undo tablespace to perform a Flashback Table operation.
- You require the FLASHBACK ANY TABLE or the FLASHBACK object privilege on the specific table.
- SELECT, INSERT, DELETE, and ALTER privileges on the table to be flashed back are required.
- You *must* enable row movement on the table that you are performing the flashback operation on.

Flashback Table

With Flashback Table, you can recover a table or tables to a specific point in time without restoring a backup. When you use this feature, the data in tables and their associated objects (indexes, constraints, triggers, and so on) is restored. The data used to satisfy a Flashback Table request is retrieved from the undo tablespace. You can use Flashback Versions Query and Flashback Transaction Query to determine the appropriate flashback time.

Flashback Table provides a way for users to easily and quickly recover from accidental modifications without a database administrator's involvement. You must grant the FLASHBACK TABLE or FLASHBACK ANY TABLE system privilege to any user that uses the Flashback Table feature. In addition, you must grant the SELECT, INSERT, DELETE, and ALTER object privileges to the user. You can use Enterprise Manager to flash back a table. The wizard guides you through the process.

Note: Enabling row movement is described on the next page.



Enabling Row Movement on a Table

Tables > Edit Table: HR.EMPLOYEES

Edit Table: HR.EMPLOYEES

Logged in as SYSTEM

Actions Create Like Go Execute On Multiple Databases Show SQL Revert Apply

General Constraints Segments Storage Options Statistics Indexes

Enable Row Movement Yes ▾

Parallel - Use multiple threads when creating this object or when executing DML against this object.
Parallel Degree Default Value

Cache - Place frequently accessed data to the top of the buffer cache.

Actions Create Like Go Execute On Multiple Databases Show SQL Revert Apply

ALTER TABLE employees ENABLE ROW MOVEMENT;

8 - 17 Copyright © 2017, BCI LTD.. All rights reserved.

Enabling Row Movement on a Table

You must enable row movement on a table to be able to flashback the table. When you enable row movement, the Oracle server can move a row in the table.

Using Enterprise Manager, you can enable row movement on a table by performing the following steps:

1. Select Tables in the Database Objects region of the Schema property page. Enter the schema name to search for the table, and click Go.
2. Click the table name of the table for which you want to enable row movement. You are now on the View Table page.
3. Click Edit, which takes you to the Edit Table page.
4. Click the Options tab, where you can change the Enable Row Movement setting for the table.
5. Set Enable Row Movement to Yes, and click Apply. The update confirmation message is displayed.



Performing Flashback Table

Point-in-time Flashback Versions Query Filter Choose SCN Flashback Tables Dependency Options Dependencies More

Perform Object Level Recovery: Point-in-time

Recovery Scope **Tables** Operation Type **Flashback Existing Tables**

Specify the point in time to which to recover.

Evaluate row changes and transactions to decide on a point in time
 * Table
 Example: SCOTT.EMP

Flashback to a timestamp
 Date Time AM PM

Flashback to a restore point
 Restore Point

Flashback to a known SCN
 SCN

Perform Recovery

Oracle Advised Recovery
 The Data Recovery Advisor detects failures in the database and presents options for performing automated repairs. Log in as SYSDBA or SYSBACKUP to use the Data Recovery Advisor.

User Directed Recovery

Recovery Scope

Operation Type Flashback Existing Tables
 Flashback Dropped Tables

```

FLASHBACK TABLE hr.departments TO TIMESTAMP
TO_TIMESTAMP('2016-04-05 21:00:00',
'YYYY-MM-DD HH24:MI:SS') ;

```

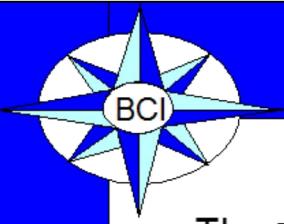
Copyright © 2017, BCI LTD.. All rights reserved.

Performing Flashback Table

You can use Enterprise Manager to flash back a table by performing the following steps:

1. Select Perform Recovery in the Backup/Recovery region on the Availability property page.
2. In the Object Level Recovery region, select Tables from the Object Type drop-down list.
3. Select Flashback Existing Tables as Operation Type. Click Recover. The “Perform Object Level Recovery: Point-in-time” page is displayed.
4. Select “Flashback to a timestamp” or “Flashback to a known SCN” and then specify a time stamp or SCN to flash back to, and click Next.
5. Click Add Tables to add tables to the list for the flashback operation. Click Next.
6. The Dependency Options page appears if there are dependent tables. Select the desired option for dealing with dependent tables. Typically, you would select “Cascade” to ensure a consistent flashback. Click Next.
7. The “Perform Object Level Recovery: Review” page appears. Review the information and click Submit. The Confirmation page appears.

Note: You can also flash back tables from the Tables link in the Schema region of the Administration page.

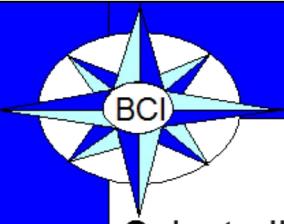


Flashback Table: Considerations

- The FLASHBACK TABLE command executes as a single transaction, acquiring exclusive DML locks.
- Statistics are not flashed back.
- Current indexes and dependent objects are maintained.
- Flashback Table operations:
 - Cannot be performed on system tables
 - Cannot span DDL operations
 - Generate undo and redo data

Flashback Table: Considerations

- The entire FLASHBACK TABLE statement is executed within a single transaction. All or none of the specified tables are flashed back.
- Flashback Table acquires exclusive data manipulation language (DML) locks on all the tables that are specified in the statement over the period of time when the operation is in progress.
- Statistics of impacted objects are not flashed back.
- All existing indexes are maintained. Dropped indexes are not re-created. Dependent on-commit materialized views are also maintained automatically.
- Tables specified in the FLASHBACK TABLE statement are flashed back, provided that none of the table constraints are violated. If any constraints are violated during flashback execution, the operation is aborted and the tables are left in the same state as they were just before the FLASHBACK TABLE statement invocation.
- You cannot perform Flashback Table to a particular time that is older than the time of the execution of a data definition language (DDL) operation that altered the structure of or shrunk a table that would be involved in the flashback operation. This restriction does not apply to DDL statements that only change storage attributes of the tables.
- Flashback Table cannot be performed on system tables, remote tables, and fixed tables.

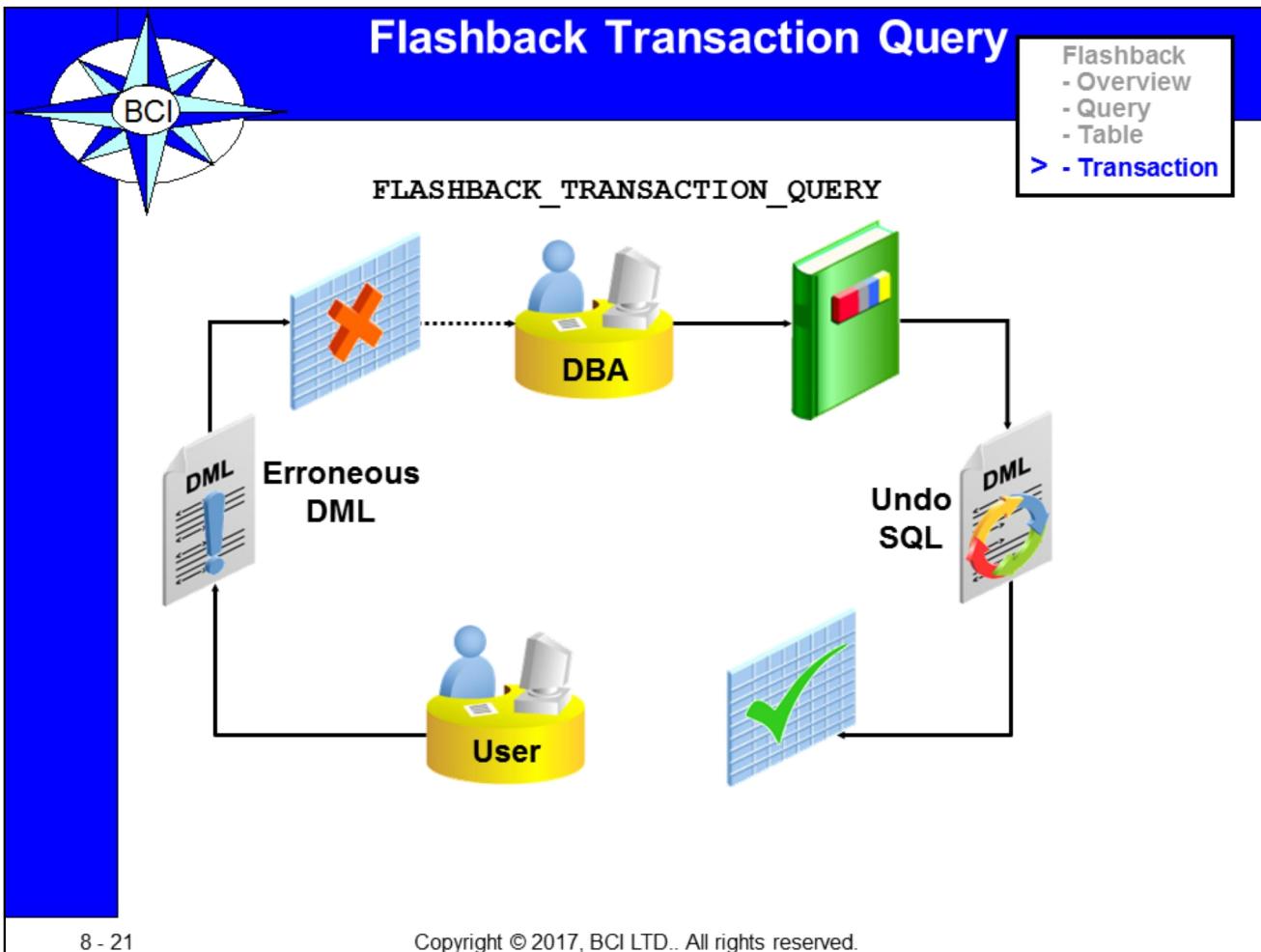


Quiz

Select all correct statements:

1. The database can remain open when a table is flashed back.
2. Flashback Table is executed as a single transaction.
3. Flashback Table requires backups to be available.
4. Flashback Table is based on undo data.

Answer: 1, 2, 4



8 - 21

Copyright © 2017, BCI LTD.. All rights reserved.

Flashback Transaction Query

Flashback Transaction Query is a diagnostic tool that you can use to view changes made to the database at the transaction level. This enables you to diagnose problems in your database and perform analysis and audits of transactions.

You can use the FLASHBACK_TRANSACTION_QUERY view to determine all the necessary SQL statements that can be used to undo the changes made either by a specific transaction or during a specific period of time.

Adding supplemental log data allows updates to be tracked with a finer level of granularity.



Using Enterprise Manager to Perform Flashback Transaction Query

The screenshot shows the Oracle Enterprise Manager interface for an Oracle Database 12c container database named ORCL12C_101-14. The user is logged in as sys. The main menu bar includes Enterprise, Targets, Favorites, History, Availability, Security, Schema, and Administration. The Availability dropdown is open, showing options like High Availability Console, MAA Advisor, Backup & Recovery (which is selected), Add Standby Database..., Schedule Backup..., Manage Current Backups, Backup Reports, Restore Points, Perform Recovery..., and Transactions.

LogMiner section:

- Query Time Range:** Set to Time Range (Start Time: Jul 27, 2015, End Time: Jul 27, 2015).
- Query Filter:** Set to View All Transactions, Container: CDB\$ROOT, Table: Examples: Scott.Emp, Scott.%.
- Transaction Results:** Shows a table with columns Transaction ID, SCN, Operation, Schema, Table, and SQL Redo. One row is highlighted:

07000E00ED070000	2916345	UPDATE	SCOTT	STAFF	update "SCOTT"."STAFF" set "COMM" = 7000 where "COMM" = 650.25 and ROWID = 'AAAWwUAAJAAAAACuAAF';
------------------	---------	--------	-------	-------	---

LogMiner Results window (highlighted with a red box):

- Summary:** Matching Transactions 1, Total Time 3 seconds. Matching Redo Records 4.
- Transaction Results:** Shows a table with columns Transaction ID, DB User, Commit Timestamp, Redo Records, and Transaction Summary - Updates (upd), Inserts (ins), Deletes (del), Other (oth). One row is highlighted:

07000E00ED070000	SYSTEM	Jul 28, 2015 3:32:04 AM	4	SCOTT.STAFF (3 upd, 1 ins)
------------------	--------	-------------------------	---	----------------------------
- TIP:** The transaction summary shows the first few tables modified by the transaction, along with the number of inserts, deletes and updates that matched the query filter.

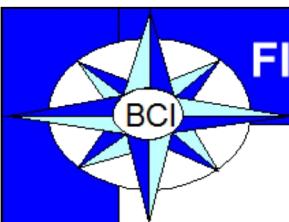
8 - 22

Copyright © 2017, BCI LTD.. All rights reserved.

Using Enterprise Manager to Perform Flashback Transaction Query

This feature is used in conjunction with the Flashback Version Query feature with the help of the Perform Recovery Wizard. On the “Perform Object Level Recovery: Choose SCN” page, click the corresponding Transaction ID link in the Flashback Version Query Result region.

In the example in the slide, a Flashback Version Query is performed on the JOBS table to retrieve the three versions of the JOBS row for `JOB_ID = 'AD_PRES'`. Then, one of the transaction IDs is clicked, showing all the changes that were part of that transaction. Notice that in addition to the JOBS table update, there was also an update to the EMPLOYEES table in that transaction.



Flashback Transaction Query: Considerations

- DDL commands are seen as dictionary updates.
- Flashback Transaction Query on a transaction underlying a DDL command displays the data dictionary changes.
- Dropped objects appear as object numbers.
- Dropped users appear as user identifiers.

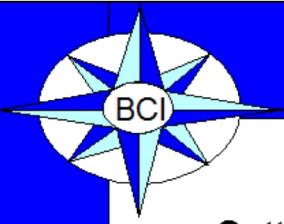
Flashback Transaction Query: Considerations

Within the database, DDL operations are nothing but a series of space management operations and changes to the data dictionary. Flashback Transaction Query on a transaction underlying a DDL command displays the changes made to the data dictionary.

When Flashback Transaction Query involves tables that have been dropped from the database, the table names are not reflected. Instead, object numbers are used.

If the user who executed a transaction is dropped, Flashback Transaction Query of that transaction displays the corresponding user ID only, and not the username.

Note: When there is not enough undo data for a specific transaction, a row with a value of UNKNOWN in the OPERATION column of FLASHBACK_TRANSACTION_QUERY is returned.



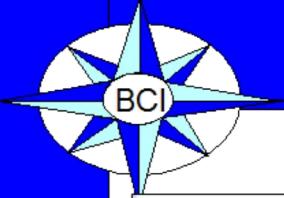
Flashback Transaction

- Setting up Flashback Transaction prerequisites
- Stepping through a possible workflow
- Using the Flashback Transaction Wizard
- Querying transactions with and without dependencies
- Choosing back-out options and flashing back transactions
- Reviewing the results

Flashback Transaction

With Flashback Transaction, you can reverse a transaction and dependant transactions. Oracle Database determines the dependencies between transactions and, in effect, creates a compensating transaction that reverses the unwanted changes. The database rewinds to a state as if the transaction, and any transactions that could be dependent on it, never occurred.

You can use the Flashback Transaction functionality from within Enterprise Manager or with PL/SQL packages.



Prerequisites

ORACLE Enterprise Manager Cloud Control 12c Database Control

Setup Preferences Help Logout Database

(X) Error

Failed in checking Flashback Transaction requirements

1. [SUPPLEMENTAL_LOG_DATA](#) - Supplemental log data is not available.
2. [EXECUTE ON DBMS_FLASHBACK](#) - User does not have execute on dbms_flashback privilege.
3. [SUPPLEMENTAL_LOG_DATA_PK](#) - Supplemental log data (primary key) is not available.
4. [SELECT ANY TRANSACTION](#) - User does not have select any transaction privilege.

Details/Required Actions

Run the following SQL command as sysdba user to correct the error

SUPPLEMENTAL_LOG_DATA: alter database add supplemental log data
SUPPLEMENTAL_LOG_DATA_PK: alter database add supplemental log data (primary key) columns
EXECUTE ON DBMS_FLASHBACK: grant execute on dbms_flashback to user.
SELECT ANY TRANSACTION: grant select any transaction to user.

SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA;
Sql> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA(PRIMARY KEY) COLUMNS

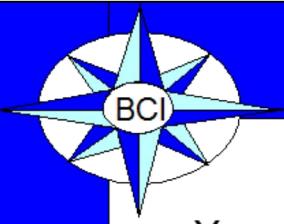
8 - 25

Copyright © 2017, BCI LTD.. All rights reserved.

Prerequisites

In order to use this functionality, supplemental logging must be enabled and the correct privileges established. For example, the HR user in the HR schema decides to use Flashback Transaction for the REGIONS table. The SYSDBA performs the following setup steps in SQL*Plus:

```
alter database add supplemental log data;
alter database add supplemental log data (primary key) columns;
grant execute on dbms_flashback to hr;
grant select any transaction to hr;
```



Flashing Back a Transaction

- You can flash back a transaction with Enterprise Manager or from the command line.
- EM uses the Flashback Transaction Wizard, which calls the `DBMS_FLASHBACK.TRANSACTION_BACKOUT` procedure with the `NOCASCADE` option.
- If the PL/SQL call finishes successfully, it means that the transaction does not have any dependencies and a single transaction is backed out successfully.



Flashing Back a Transaction

Security privileges

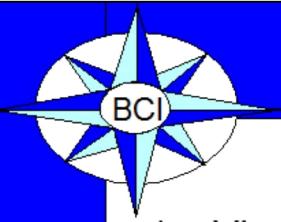
To flash back or back-out a transaction—that is, to create a compensating transaction—you must have the `SELECT`, `FLASHBACK`, and `DML` privileges on all affected tables.

Conditions of Use

- Transaction back-out is not supported across conflicting DDL.
- Transaction back-out inherits data type support from LogMiner. See the Oracle Database 12c documentation for supported data types.

Recommendation

- When you discover the need for transaction back-out, performance is better if you start the back-out operation sooner. Large redo logs and high transaction rates result in slower transaction back-out operations.
- Provide a transaction name for the back-out operation to facilitate later auditing. If you do not provide a transaction name, it will be automatically generated for you.



Possible Workflow

1. Viewing data in a table
2. Discovering a logical problem
3. Using Flashback Transaction
 1. Performing a query
 2. Selecting a transaction
 3. Flashing back a transaction (with no conflicts)
 4. Choosing other back-out options (if conflicts exists)
4. Reviewing Flashback Transaction results



Possible Workflow

Assume that several transactions occurred as indicated below:

```
connect hr
Enter password: oracle_4U <<< not displayed
INSERT INTO hr.regions VALUES (5,'Pole');
COMMIT;
UPDATE hr.regions SET region_name='Poles' WHERE region_id = 5;
UPDATE hr.regions SET region_name='North and South Poles' WHERE region_id
= 5;
COMMIT;
INSERT INTO hr.countries VALUES ('TT','Test Country',5);
COMMIT;
connect sys/<password> as sysdba
ALTER SYSTEM ARCHIVE LOG CURRENT;
```

8 - 28

Copyright © 2017, BCI LTD.. All rights reserved.

Flashback Transaction Wizard

In Enterprise Manager, select Schema > Database Objects > Table. Then select SCOTT.STAFF under Table, select Flashback Transaction from the Actions drop-down list, and then click Go. This invokes the Flashback Transaction Wizard for your selected table. The Flashback Transaction: Perform Query page is displayed.

Select the appropriate time range and add query parameters. (The more specific you can be, the shorter is the search of the Flashback Transaction Wizard.)

Without Enterprise Manager, use the DBMS_FLASHBACK.TRANSACTION_BACKOUT procedure, which is described in the *PL/SQL Packages and Types Reference*. Essentially, you take an array of transaction IDs as the starting point of your dependency search. For example:

```

CREATE TYPE XID_ARRAY AS VARRAY(100) OF RAW(8);
CREATE OR REPLACE PROCEDURE TRANSACTION_BACKOUT(
    numberOFTXIDs NUMBER, -- number of transactions passed as input
    xids XID_ARRAY, -- the list of transaction ids
    options NUMBER default NOCASCADE, -- back out dependent
    txn timeHint TIMESTAMP default MINTIME -- time hint on the txn
    start
);

```



Choosing Other Back-out Options

Flashback Transaction: Select Transaction

Database **pec1**
Operation Type **Flashback Transaction**

Select a transaction and click on Next to flashback the transaction. You will have a chance to examine the dependencies and confirm the operation.

Query Results

Select	Transaction ID	DB User	Commit Timestamp ▲	Redo Records	Transaction Summary - Updates (upd), Inserts (ins), Deletes (del), Other (oth)
<input checked="" type="checkbox"/>	07001100FF050000	SCOTT	Aug 3, 2015 9:17:18 AM	35	SCOTT.STAFF (35 ins)
<input checked="" type="checkbox"/>	0A000E00FA060000	SCOTT	Aug 3, 2015 9:30:34 AM	6	SCOTT.STAFF (5 upd, 1 del)

Transaction Details

SCN ▲	Operation	Schema	Table	SQL Redo
2270943	START			set transaction read write;
2270943	UPDATE	SCOTT	STAFF	update "SCOTT"."STAFF" set "COMM" = 7000 where "ID" = 330 and "NAME" = 'BURKE' and "DEPT" = 66 and "JOB" = 'CLERK' and "YEARS" = 1 and "SALARY" = 10988 and "COMM" = 55.5 and ROWID = 'AAAI\cnAAGAAAAFgAag';
2270943	UPDATE	SCOTT	STAFF	update "SCOTT"."STAFF" set "COMM" = 7000 where "ID" = 340 and "NAME" = 'EDWARDS' and "DEPT" = 64 and "JOB" = 'SALES' and "YEARS" = 7 and "SALARY" = 17844 and "COMM" = 1285 and ROWID = 'AAAI\cnAAGAAAAFgAAH';
2270943	UPDATE	SCOTT	STAFF	update "SCOTT"."STAFF" set "COMM" = 7000 where "ID" = 350 and "NAME" = 'GAFNEY' and "DEPT" = 64 and "JOB" = 'CLERK' and "YEARS" = 5 and "SALARY" = 13030.5 and "COMM" = 188 and ROWID = 'AAAI\cnAAGAAAAFgAAU';
2270998	DELETE	SCOTT	STAFF	delete from "SCOTT"."STAFF" where "ID" = 50 and "NAME" = 'HANES' and "DEPT" = 15 and "JOB" = 'MGR' and "YEARS" = 10 and "SALARY" = 20659.8 and "COMM" IS NULL and ROWID = 'AAAI\cnAAGAAAAFgAAE';
2271004	UPDATE	SCOTT	STAFF	update "SCOTT"."STAFF" set "SALARY" = 70000 where "ID" = 340 and "NAME" = 'EDWARDS' and "DEPT" = 64 and "JOB" = 'SALES' and "YEARS" = 7 and "SALARY" = 17844 and "COMM" = 7000 and ROWID = 'AAAI\cnAAGAAAAFgAAH';
2271009	UPDATE	SCOTT	STAFF	update "SCOTT"."STAFF" set "JOB" = 'MGR' where "ID" = 330 and "NAME" = 'BURKE' and "DEPT" = 66 and "JOB" = 'CLERK' and "YEARS" = 1 and "SALARY" = 10988 and "COMM" = 7000 and ROWID = 'AAAI\cnAAGAAAAFgAag';
2271021	COMMIT			commit;

1 **2**

Copyright © 2017, BCI LTD.. All rights reserved.

8 - 29

Choosing Other Back-out Options

The TRANSACTION_BACKOUT procedure checks dependencies, such as:

- Write-after-write (WAW)
- Primary and unique constraints
- Foreign key constraints

A transaction can have a WAW dependency, which means a transaction updates or deletes a row that has been inserted or updated by a dependent transaction. This can occur, for example, in a master/detail relationship of primary (or unique) and mandatory foreign key constraints.

To understand the difference between the NONCONFLICT_ONLY and the NOCASCADE_FORCE options, assume that the T1 transaction changes rows R1, R2, and R3 and the T2 transaction changes rows R1, R3, and R4. In this scenario, both transactions update row R1, so it is a “conflicting” row. The T2 transaction has a WAW dependency on the T1 transaction. With the NONCONFLICT_ONLY option, R2 and R3 are backed out, because there is no conflict and it is assumed that you know best, what to do with the R1 row. With the NOCASCADE_FORCE option, all three rows (R1, R2, and R3) are backed out.

Note: This screenshot is not part of the workflow example, but shows additional details of a more complex situation.

Choosing Other Back-out Options

Step 1. Choose Columns

Available Columns: ID, NAME, DEPT, YEARS
Selected Columns: SALARY, COMM, JOB

Step 2. Bind The Row Value

Specify a where clause based on the columns selected above to narrow the search to a particular set of values.

Example: where JOB='CLERK'

Step 3. Select Time Interval

Show All Row History (radio button selected)
Specify Timestamp

Start Date: Aug 3, 2015
End Date: Aug 3, 2015
Start Time: 09:31 AM
End Time: 09:50 AM

Flashback Versions Query Result

Select	Flashback SCN	Flashback Timestamp	Transaction ID	Operation	SALARY	COMM	JOB
<input checked="" type="radio"/>	2271020	Aug 3, 2015 9:30:33 AM	0A000E00FA060000	UPDATE	10988	7000	MGR
<input type="radio"/>	2271020	Aug 3, 2015 9:30:33 AM	0A000E00FA060000	UPDATE	13030.5	7000	CLERK
<input type="radio"/>	2271020	Aug 3, 2015 9:30:33 AM	0A000E00FA060000	UPDATE	70000	7000	SALES

Flashback Versions Query Filter: Flashback Versions Query SQL

```
select versions_startscn, to_char(versions_starttime, 'YYYY-MM-DD HH24:MI:SS'), versions_xid, versions_operation, SALARY, COMM, JOB from SCOTT.STAFF versions
between scn minvalue and maxvalue where versions_startscn is not null order by versions_startscn desc
```

Choosing Other Back-out Options (continued)

The Flashback Transaction Wizard works as follows:

If the DBMS_FLASHBACK.TRANSACTION_BACKOUT procedure with the NOCASCADE option fails (because there are dependent transactions), you can change the recovery options.

- With the Nonconflict Only option, nonconflicting rows within a transaction are backed out, which implies that database consistency is maintained (although the transaction atomicity is broken for the sake of data repair).
- If you want to forcibly back out the given transactions, without paying attention to the dependent transactions, use the Nocascade Force option. The server simply executes the compensating DML commands for the given transactions in reverse order of their commit times. If no constraints break, you can proceed to commit the changes, or else roll back.
- To initiate the complete removal of the given transactions and all their dependents in a post-order fashion, use the Cascade option.

Note: This screenshot is not part of the workflow example, but shows additional details of a more complex situation.



Final Steps in EM Backup REcovery

Perform Object Level Recovery: Flashback Tables

Recovery Scope: Tables
Operation Type: Flashback Existing Tables

Your application may have tables that are logically related to this table. Specify all such tables that must be flashed back to the SCN you selected.

Evaluated Table Name	SCOTT.STAFF
Flashback Time	Aug 3, 2015 09:30 AM
Flashback SCN	2271020
Tables To Flashback	SCOTT.STAFF <input type="button" value="Add Tables"/>

Example: scott.emp, one table name per row

Cancel Back Step 4 of 7 Next

4

Perform Object Level Recovery: Review

Recovery Scope: Tables
Operation Type: Flashback Existing Tables

The following tables will be flashed back. These tables will be locked while the flashback operation is in progress.

SCN	2271020
Timestamp	Aug 3, 2015 09:30 AM
Table	SCOTT.STAFF
Dependent Tables	

Cancel Show Row Changes Show SQL Back Step 7 of 7 Submit

5

After choosing your back-out option, the dependency report is generated in the DBA_FLASHBACK_TXN_STATE and DBA_FLASHBACK_TXN_REPORT views.

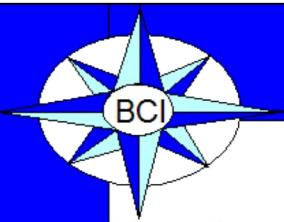
- Review the dependency report that shows all transactions which were backed out.
- Commit the changes to make them permanent.
- Roll back to discard the changes.

Final Steps Without EM

The DBA_FLASHBACK_TXN_STATE view contains the current state of a transaction: whether it is alive in the system or effectively backed out. This table is atomically maintained with the compensating transaction. For each compensating transaction, there could be multiple rows, where each row provides the dependency relationship between the transactions that have been compensated by the compensating transaction.

The DBA_FLASHBACK_TXN_REPORT view provides detailed information about all compensating transactions that have been committed in the database. Each row in this view is associated with one compensating transaction.

For a detailed description of these tables, see the *Oracle Database Reference*.



Quiz

You discover that Jim's salary was updated twice. The first update was correct, but the second was done by mistake. Before you discovered this problem, other rows in the EMPLOYEES table are updated correctly. Which technology should you use to fix this error?

1. Flashback Database
2. Flashback Query
3. Flashback Transaction

Answer: 3



Summary

In this lesson, you should have learned how to:

- Describe Flashback technology
- Perform Flashback Query
- Use Flashback Version Query
- Enable row movement on a table
- Perform Flashback Table operations
- Use Flashback Transaction Query
- Use Flashback Transaction



Practice 8: Performing Flashback Transaction Backout

This practice covers the following topics:

- Querying a transaction
- Performing Flashback Transaction backout