

Data Selection and Filtering

(Part I)

by Ahmed Baraka

Introduction to Oracle Data Guard

In this lecture, we are going to talk about the basic concepts of Oracle Data Guard

Objectives

By the end of this lecture, you should be able to:



- Set up a data selection based on a table, rows, operations and columns
- Select rows with `WHERE` Clause
- Select rows with `FILTER` Clause
- Ensure data availability for filters
- Perform columns mapping

Data Filtering and Manipulation Overview

- Table, rows, operation, and columns selection
- Data manipulation and transformation



Limitation on Data Filtering and Manipulation

- Oracle GoldenGate does not support the filtering, column mapping, or manipulation on large objects (LOB).

Data Selection (Filtering)

- Table Selection
 - `TABLE (Extract)` or `MAP (Replicat)` parameters
- Rows Selection
 - `Where` or `Filter` clause
- Operations Selection
 - `Filter` clause
- Columns Selection
 - `Colmap`, `Cols` or `ColsExcept`




Selecting Rows with WHERE Clause

- For basic selection
- Must be enclosed in parentheses
- Arithmetic operations are not supported



Supported Evaluation	Examples
Column names	EMPLOYEE_ID, EMPLOYEE_NAME
Operators	=, <>, >, <, >=, <=, AND, OR
Numeric values	-45, 23.6
Literals	'COURSE', 'Wa'
Column test	@NULL, @PRESENT, @ABSENT

Column Tests

Column Test	Description	 Ahmed Baraka Oracle Database Administrator
@NULL	To test if the column value is null	
@PRESENT	To test if the column value is present in the current operation	
@ABSENT	To test if the column value is absent in the current operation	

WHERE Clause Examples

```
Map HR.EMPLOYEES, Target UAE.EMPLOYEES,  
  where (LOC_CODE = "UAE");
```



```
... where (DOB = @NULL);
```

```
Map HR.EMPLOYEES, Target UAE.EMPLOYEES,  
  where (SALARY = @PRESENT and SALARY >= 10000);
```


Selecting Rows with **FILTER** Clause

- For complex *numeric* evaluation
- GoldenGate column-conversion functions are supported
- User-defined error is supported
- Multiple FILTER clause is supported



```
MAP source_table, TARGET target_table,  
  , FILTER (  
    [, ON INSERT | ON UPDATE | ON DELETE]  
    [, IGNORE INSERT | IGNORE UPDATE | IGNORE DELETE]  
    [, RAISEERROR error_number]  
    , filter_clause);
```

Valid **FILTER** Clause Elements

- Oracle GoldenGate column-conversion functions
- Numbers and the columns that contain numbers
- Arithmetic operators: **+**, **-**, *****, **/** (divide), **** (remainder)
- Comparison operators: **>**, **>=**, **<**, **<=**, **=**, **<>**
- Logical operators: **AND**, **OR**
- Non-numeric evaluation is not supported: **AMOUNT=@PRESENT**



FILTER Clause Examples



```
Filter ( @COMPUTE (UNIT_PRICE * QUANTITY) > 1000);  
Filter ( @StrFind(NAME, "BARAKA") > 0);  
Filter ( ON UPDATE, ON DELETE,  
        @Compute (UNIT_PRICE * QUANTITY) > 1000);
```

```
Filter ( ON UPDATE, SALARY > 1000 ), where( SALARY =  
@PRESENT )
```

Ensuring Data Availability for Filters

- Use only primary-key columns as selection criteria
- Enable supplemental logging for the columns
- Use the FETCHCOLS or FETCHCOLSEXCEPT option of the TABLE parameter



```
TABLE HR.EMPLOYEES, FETCHBEFOREFILTER, FETCHCOLS (age),  
FILTER (age > 50);
```

- Test for a column's presence first, then for the column's value

```
where (SALARY = @PRESENT and SALARY >= 10000);
```

Mapping Columns

- Usually required when tables have different data structures
- Table-level (COLMAP)
 - Map individual source columns to target columns

```
COLMAP ([USEDEFAULTS,] <target_col> = <source_expr>)
```

- Global level (COLMATCH)
 - Global rules for column mapping
- USEDEFAULTS: automatically maps source and target columns

COLMAP Clause Example: Column Mapping

- Column Mapping and Transformation example:



```
Map HR.CUSTOMER, Target HR.CONTACT,  
ColMap (USEDEFAULTS,  
    CUST_GENDER = GENDER,  
    FULL_NAME = @StrCat( LAST_NAME, " ",  
                          MIDDLE_NAME, ",", LAST_NAME ) );
```

COLMAP Clause Example: Building history

- Building history



```
...
InsertAllRecords
Map SALES.INVOICE, Target REPORT.INV_HISTORY,
  ColMap (USEDEFAULTS,
    TRAN_TIME = @GetEnv("GGHEADER", "COMMITTIMESTAMP"),
    OP_TYPE = @GetEnv("GGHEADER", "OPTYPE"),
    BA_IND = @GetEnv("GGHEADER", "BEFOREAFTERINDICATOR")
  );
```

COLMATCH Clause Example

```
COLMATCH NAMES EMPLOYEE_ID = EMP_NO  
COLMATCH NAMES CUSTOMER_NAME = CUST_NAME  
COLMATCH NAMES LEAVE_ID = LEAVE_SN  
MAP HR.EMPLOYEES, TARGET HR.EMP, COLMAP (USEDEFAULTS);  
MAP HR.LEAVES, TARGET HR.LVS, COLMAP (USEDEFAULTS);  
COLMATCH RESET  
MAP SALES.REGION, TARGET SALE.REG;  
MAP SALES.PRICE, TARGET SALES.PRICE;
```



Retrieving Before and After Values

- To reference the before value:



1. Use the `LOGALLSUPCOLS` parameter in the Extract.

Note: for Oracle GoldenGate of releases prior to 12c, use the `GETUPDATEBEFORES` parameter.

2. Use the `@BEFORE` column conversion function

```
COLMAP (... , DELTA = @COMPUTE( BALANCE - @BEFORE(BALANCE))
);
```

Summary

In this lecture, you should have learnt to:



- Set up a data selection based on a table, rows, operations and columns
- Select rows with WHERE Clause
- Select rows with FILTER Clause
- Ensure data availability for filters
- Perform columns mapping