## Task 01: Create Packages for Reload Dimension from SA\_\*

Create Packages to reload Dim data

* + Use Explicit Cursor (One package)

CREATE OR REPLACE PACKAGE BODY pkg\_etl\_dim\_customers\_dw

AS

*-- Load Data From Sources table to DataBase*

PROCEDURE load\_tmp\_customers

AS

CURSOR c\_fg

IS

( SELECT DISTINCT sa\_t.cust\_first\_name

, t2.geo\_id

, sa\_t.cust\_last\_name

, sa\_t.cust\_gender

, sa\_t.cust\_birth\_year

, sa\_t.cust\_email

, sa\_t.cust\_pass\_number

, sa\_t.cust\_balance

, sa\_t.cust\_level\_income

, t3.cust\_pass\_number AS code

, t3.cust\_balance AS bal

, t3.cust\_level\_income AS lev

FROM u\_sa\_data.tmp\_customers sa\_t

LEFT OUTER JOIN u\_dw\_references.cu\_countries t2

ON ( sa\_t.cust\_country\_desc = t2.region\_desc )

LEFT JOIN u\_dw.dw\_customers t3

ON ( sa\_t.cust\_pass\_number = t3.cust\_pass\_number ) );

BEGIN

DELETE FROM u\_dw.dw\_customers dw\_t

WHERE dw\_t.cust\_pass\_number NOT IN (SELECT DISTINCT cust\_pass\_number

FROM u\_sa\_data.tmp\_customers);

*--Insert Source data*

FOR i IN c\_fg LOOP

IF i.cust\_pass\_number = i.code THEN

UPDATE u\_dw.dw\_customers dw\_c

SET dw\_c.cust\_balance = i.cust\_balance

, dw\_c.cust\_level\_income = i.cust\_level\_income

, dw\_c.update\_dt = SYSDATE

WHERE i.cust\_pass\_number = i.code

AND ( i.cust\_balance != i.bal

OR i.cust\_level\_income != i.lev );

ELSIF i.cust\_pass\_number != NVL ( i.code, -99 ) THEN

INSERT INTO u\_dw.dw\_customers ( cust\_id

, cust\_geo\_id

, cust\_first\_name

, cust\_last\_name

, cust\_gender

, cust\_year\_of\_birth

, cust\_email

, cust\_pass\_number

, cust\_balance

, cust\_level\_income

, insert\_dt )

VALUES ( seq\_customers.NEXTVAL

, i.geo\_id

, i.cust\_first\_name

, i.cust\_last\_name

, i.cust\_gender

, i.cust\_birth\_year

, i.cust\_email

, i.cust\_pass\_number

, i.cust\_balance

, i.cust\_level\_income

, SYSDATE );

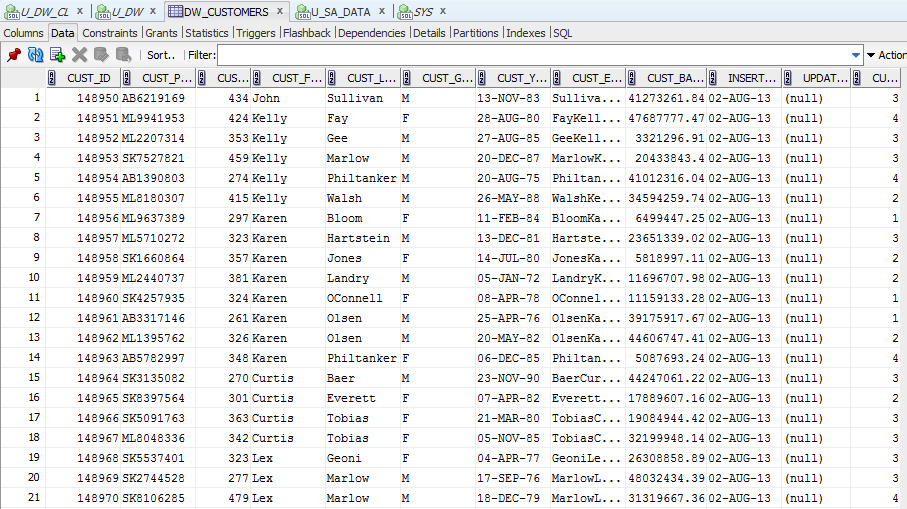
END IF;

END LOOP;

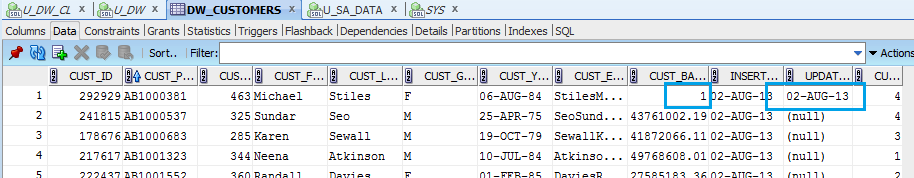
COMMIT;

END load\_tmp\_customers;

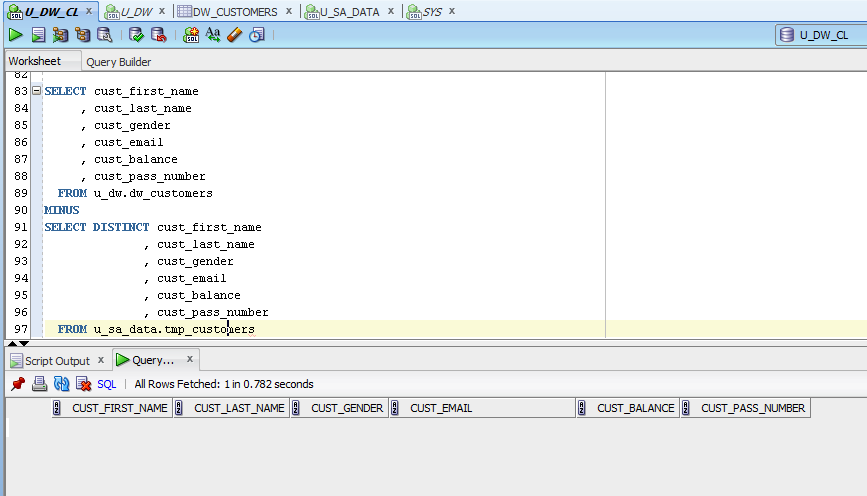
END pkg\_etl\_dim\_customers\_dw;



After updating the data in the source table and restart the package data is changed at stage-level.



After repeated run the package the data in the table are unchanged. Test results are shown below:



* + Use Merge (One packages)

CREATE OR REPLACE PACKAGE BODY pkg\_etl\_dim\_operations\_dw

AS

*-- Reload Data From Sources table to DataBase*

PROCEDURE load\_tmp\_methods

AS

BEGIN

*--Merge Source data*

MERGE INTO u\_dw.dw\_operation\_methods o\_m

USING (SELECT DISTINCT operation\_method\_id

, operation\_method\_name

, operation\_method\_type

, operation\_method\_type\_id

FROM u\_sa\_data.tmp\_methods) cls

ON ( o\_m.operation\_method\_code = cls.operation\_method\_id )

WHEN NOT MATCHED THEN

INSERT ( o\_m.operation\_method\_id

, o\_m.operation\_method\_code

, o\_m.operation\_method\_name

, o\_m.operation\_method\_type

, o\_m.operation\_method\_type\_id

, o\_m.insert\_dt )

VALUES ( seq\_operation\_methods.NEXTVAL

, cls.operation\_method\_id

, cls.operation\_method\_name

, cls.operation\_method\_type

, cls.operation\_method\_type\_id

, SYSDATE )

WHEN MATCHED THEN

UPDATE SET o\_m.operation\_method\_name = cls.operation\_method\_name

, o\_m.operation\_method\_type = cls.operation\_method\_type

, o\_m.operation\_method\_type\_id = cls.operation\_method\_type\_id

, o\_m.update\_dt = SYSDATE

WHERE o\_m.operation\_method\_type != cls.operation\_method\_type

OR o\_m.operation\_method\_type\_id != cls.operation\_method\_type\_id

OR o\_m.operation\_method\_name != cls.operation\_method\_name;

*--Commit Result*

COMMIT;

END load\_tmp\_methods;

PROCEDURE load\_tmp\_operations

AS

BEGIN

*--Merge Source data*

MERGE INTO u\_dw.dw\_operations o\_s

USING (SELECT DISTINCT operation\_id

, operation\_name

, operation\_max\_amount

, operation\_min\_amount

FROM u\_sa\_data.tmp\_operations) cls

ON ( o\_s.operation\_code = cls.operation\_id )

WHEN NOT MATCHED THEN

INSERT ( operation\_id

, operation\_code

, operation\_name

, operation\_max\_amount

, operation\_min\_amount

, insert\_dt )

VALUES ( seq\_operations.NEXTVAL

, cls.operation\_id

, cls.operation\_name

, cls.operation\_max\_amount

, cls.operation\_min\_amount

, SYSDATE )

WHEN MATCHED THEN

UPDATE SET o\_s.operation\_name = cls.operation\_name

, o\_s.operation\_max\_amount = cls.operation\_max\_amount

, o\_s.operation\_min\_amount = cls.operation\_min\_amount

, o\_s.update\_dt = SYSDATE

WHERE o\_s.operation\_name != cls.operation\_name

OR o\_s.operation\_max\_amount != cls.operation\_max\_amount

OR o\_s.operation\_min\_amount != cls.operation\_min\_amount

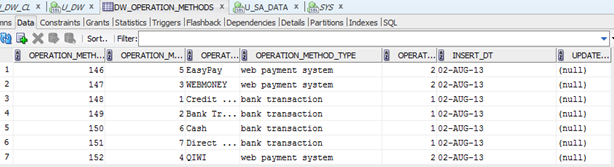
;

*--Commit Result*

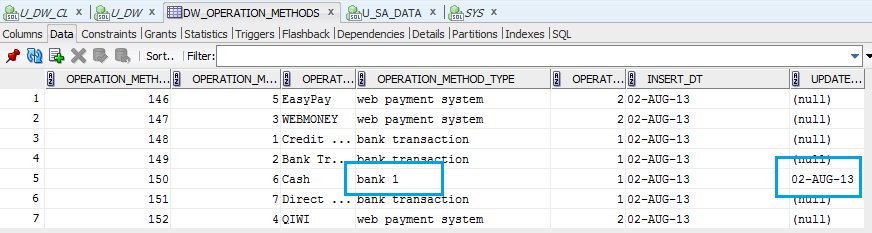
COMMIT;

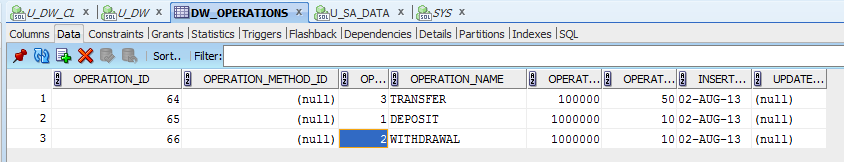
END load\_tmp\_operations;

END pkg\_etl\_dim\_operations\_dw;

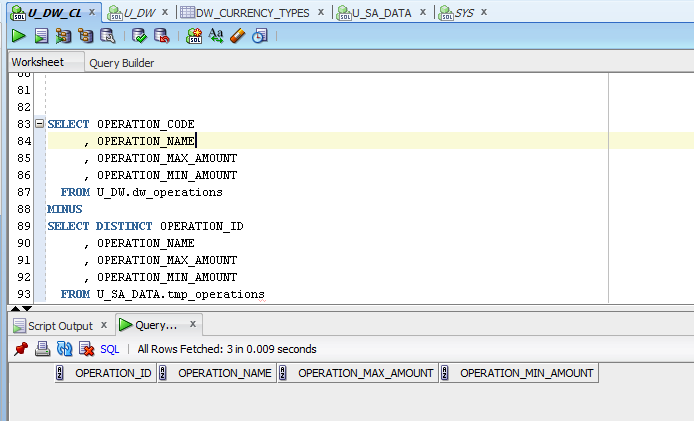


After updating the data in the source table and restart the package data is changed at stage-level.





After repeated run the package the data in the table are unchanged. Test results are shown below:



* + Use Explicit Cursor and Bulk Insertion

CREATE OR REPLACE PACKAGE BODY pkg\_etl\_currency\_dw

AS

*-- Reload Data From Sources table to DataBase*

PROCEDURE load\_currency\_types

AS

CURSOR c1

IS

SELECT DISTINCT sa\_t.currency\_type\_id

, sa\_t.currency\_type\_name

, dw\_t.currency\_type\_desc

, dw\_t.currency\_type\_code

FROM u\_sa\_data.tmp\_currency sa\_t

LEFT JOIN

u\_dw.dw\_currency\_types dw\_t

ON ( dw\_t.currency\_type\_code = sa\_t.currency\_type\_id );

TYPE cur\_type IS TABLE OF c1%ROWTYPE;

curr cur\_type;

BEGIN

OPEN c1;

FETCH c1

BULK COLLECT INTO curr;

FOR i IN curr.FIRST .. curr.LAST LOOP

IF curr ( i ).currency\_type\_id = curr ( i ).currency\_type\_code THEN

UPDATE u\_dw.dw\_currency\_types ct

SET ct.currency\_type\_desc = curr ( i ).currency\_type\_name

, update\_dt = SYSDATE

WHERE curr ( i ).currency\_type\_id = ct.currency\_type\_code

AND curr ( i ).currency\_type\_name != ct.currency\_type\_desc;

ELSIF curr ( i ).currency\_type\_id != NVL ( curr ( i ).currency\_type\_code, -99 ) THEN

INSERT INTO u\_dw.dw\_currency\_types ( currency\_type\_id

, currency\_type\_code

, currency\_type\_desc

, insert\_dt )

VALUES ( seq\_currency\_type.NEXTVAL

, curr ( i ).currency\_type\_id

, curr ( i ).currency\_type\_name

, SYSDATE );

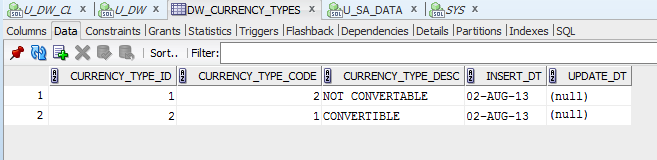
END IF;

END LOOP;

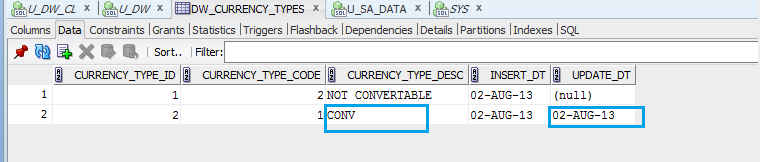
COMMIT;

END load\_currency\_types;

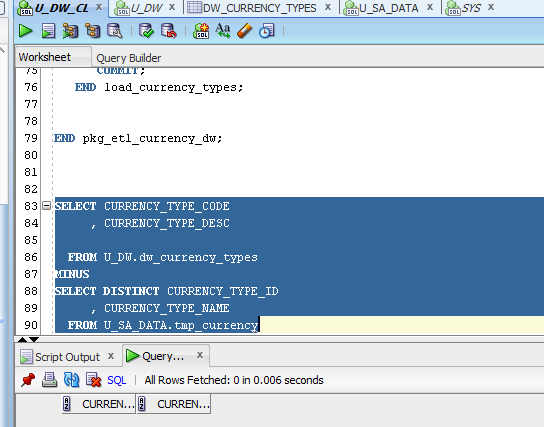
END pkg\_etl\_currency\_dw;



After updating the data in the source table and restart the package data is changed at stage-level.



After repeated run the package the data in the table are unchanged. Test results are shown below:



* + Use Variable Cursor and FORALL Bulk Insertion (One package)

CREATE OR REPLACE PACKAGE BODY pkg\_etl\_operations\_dw

AS

*-- Load Data From Sources table to DataBase*

PROCEDURE load\_operations

AS

TYPE cur IS REF CURSOR;

TYPE t\_id IS TABLE OF NUMBER;

TYPE t\_code IS TABLE OF NUMBER;

TYPE t\_name IS TABLE OF VARCHAR2 ( 50 );

TYPE t\_method\_id IS TABLE OF NUMBER;

TYPE t\_min IS TABLE OF NUMBER;

TYPE t\_max IS TABLE OF NUMBER;

c1 cur;

blk\_id t\_id;

blk\_code t\_code;

blk\_name t\_name;

blk\_min t\_min;

blk\_max t\_max;

BEGIN

*--Insert Source data*

OPEN c1 FOR

SELECT DISTINCT sa\_t.operation\_id

, dw\_t.operation\_code

, sa\_t.operation\_name

, sa\_t.operation\_min\_amount

, sa\_t.operation\_max\_amount

FROM u\_sa\_data.tmp\_operations sa\_t

LEFT OUTER JOIN

u\_dw.dw\_operations dw\_t

ON ( dw\_t.operation\_code = sa\_t.operation\_id );

LOOP

FETCH c1

BULK COLLECT INTO blk\_id, blk\_code, blk\_name, blk\_min, blk\_max

LIMIT 1000;

FORALL i IN blk\_code.FIRST .. blk\_code.LAST

MERGE INTO u\_dw.dw\_operations o\_s

USING (SELECT \*

FROM u\_sa\_data.tmp\_operations) cls

ON ( o\_s.operation\_code = cls.operation\_id )

WHEN NOT MATCHED THEN

INSERT ( operation\_id

, operation\_code

, operation\_name

, operation\_max\_amount

, operation\_min\_amount

, insert\_dt )

VALUES ( seq\_operations.NEXTVAL

, blk\_id ( i )

, blk\_name ( i )

, blk\_max ( i )

, blk\_min ( i )

, SYSDATE )

WHEN MATCHED THEN

UPDATE SET o\_s.operation\_name = cls.operation\_name

, o\_s.operation\_max\_amount = cls.operation\_max\_amount

, o\_s.operation\_min\_amount = cls.operation\_min\_amount

, o\_s.update\_dt = SYSDATE

WHERE o\_s.operation\_name != cls.operation\_name

OR o\_s.operation\_max\_amount != cls.operation\_max\_amount

OR o\_s.operation\_min\_amount != cls.operation\_min\_amount;

EXIT WHEN c1%NOTFOUND;

END LOOP;

CLOSE c1;

DELETE FROM u\_dw.dw\_operations

WHERE operation\_id IN (SELECT t.operation\_id

FROM u\_dw.dw\_operations t

WHERE t.ROWID > (SELECT MIN ( tt.ROWID )

FROM u\_dw.dw\_operations tt

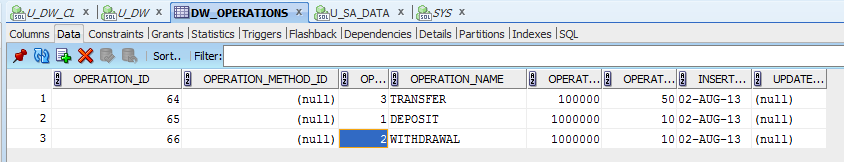
WHERE t.operation\_name = tt.operation\_name));

*--Commit Result*

COMMIT;

END load\_operations;

END pkg\_etl\_operations\_dw;



After repeated run the package the data in the table are unchanged. Test results are shown below:

