|  |
| --- |
| EPAM Systems, RD Dep. |
| Core PL SQL |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| REVISION HISTORY | | | | | |
| Ver. | Description of Change | Author | Date | Approved | |
| Name | Effective Date |
| 1.0 | Initial status | Arina Marchenko | 4-DEC-2017 |  |  |

Contents

[1. Tables 3](#_Toc500193888)

[2. Using cursors 6](#_Toc500193889)

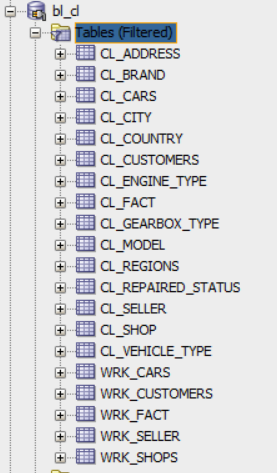
[3. Using merge 7](#_Toc500193890)

# Tables

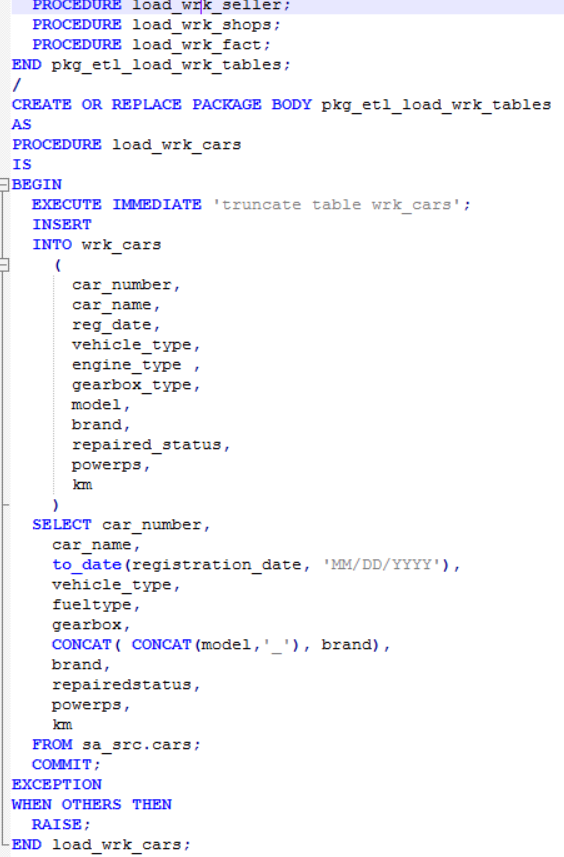
On BL\_CL layer there are two types of tables: wrk and cls.

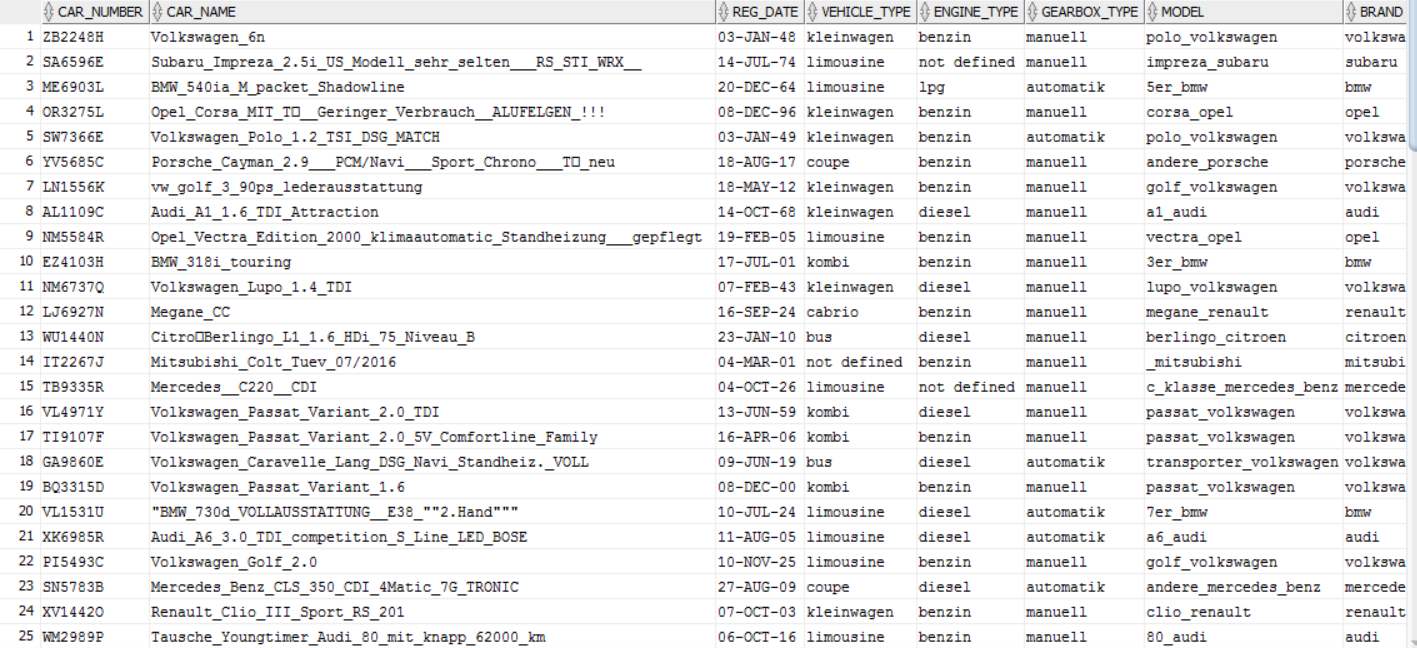
WRK tables were needed for loading external tables because external tables need a lot of space and work worse. Because of that we made a WRK tables which had the same structure as external tables, but their performance higher.

CLS tables were made for 3nf scheme, because they have the same structure as tables from 3nf, and they were made for cleansing some unsuitable information and divided some information for using in 3nf.



For loading wrk tables I used script below





Also in this tables I used only the columns, which I need. So this scrips become a little bit more complicated. Moreover, I made such stuff, like -99, -98 and not defined for null values. So on this layer I have already replace nulls.

On CLS Layer I divide WRK tables into tables that I will have on 3nf layer. Also I got rid of repetitions and generate some date columns where needed.

For loading data from wrk to cls I used only implicit cursor.

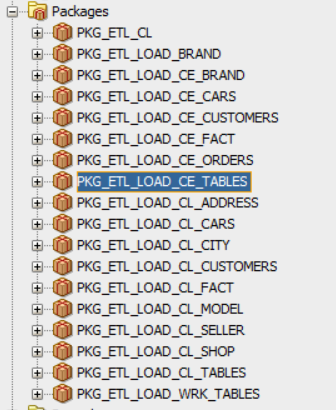


The script above was used for loading data from CLS to CE layer. Here I used merge for defining whether we already has this information or its updating. Here I also have date of last update.

Moreover on this layer I generated surr key which is primary key in CE for linking tables.

For loading data from cls to ce I used implicit and explicit cursors.

Packages, which were used for inserting data into ce are below:



# Using cursors

Here is the script for explicit cursor:

CREATE OR REPLACE PACKAGE pkg\_etl\_load\_ce\_brand

AS

PROCEDURE load\_ce\_brand;

END pkg\_etl\_load\_ce\_brand;

/

CREATE OR REPLACE PACKAGE BODY pkg\_etl\_load\_ce\_brand

AS

PROCEDURE load\_ce\_brand

IS

CURSOR c\_data

IS

SELECT DISTINCT cebr.rowid AS rid,

clbr.brand\_name,

trunc(sysdate) update\_dt

FROM cl\_brand clbr

LEFT OUTER JOIN bl\_3nf.ce\_brand cebr

ON clbr.brand\_name <> cebr.brand\_name;

type t\_\_data

IS

TABLE OF c\_data%rowtype INDEX BY binary\_integer;

t\_data t\_\_data;

BEGIN

EXECUTE IMMEDIATE 'truncate table bl\_3nf.ce\_brand';

OPEN c\_data;

LOOP

FETCH c\_data bulk collect INTO t\_data ;

EXIT

WHEN t\_data.count = 0;

FOR idx IN t\_data.first .. t\_data.last

LOOP

IF t\_data(idx).rid IS NULL THEN

INSERT

INTO bl\_3nf.ce\_brand

(

brand\_id,

brand\_name,

update\_dt

)

VALUES

(

bl\_3nf.seq\_brand.nextval,

t\_data(idx).brand\_name,

sysdate

);

ELSE

UPDATE bl\_3nf.ce\_brand

SET update\_dt = sysdate

WHERE rowid = t\_data(idx).rid ;

END IF;

END LOOP;

END LOOP;

CLOSE c\_data;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

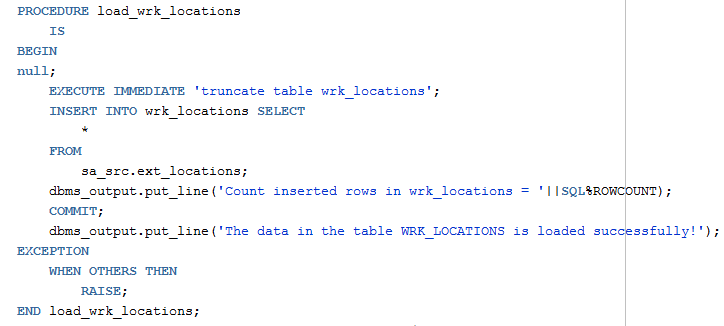
RAISE;

END load\_ce\_brand;

END pkg\_etl\_load\_ce\_brand;

/

Implicit cursor are used while INSERT, UPDATE or SELECT INTO. You don’t need either declare them or open



# Using merge

Here is the script for loading cars table from cls to ce.

PROCEDURE load\_ce\_cars

IS

BEGIN

EXECUTE IMMEDIATE 'truncate table bl\_3nf.ce\_cars';

MERGE INTO bl\_3nf.ce\_cars cc

USING ( SELECT car\_number, car\_name, vehicle\_type\_id, engine\_type\_id,

gearbox\_type\_id, model\_id, repaired\_status\_id, start\_dt, end\_dt FROM cl\_cars ) clc

ON (cc.car\_number = clc.car\_number and cc.start\_dt = clc.start\_dt and

cc.end\_dt = clc.end\_dt )

WHEN MATCHED THEN

UPDATE SET cc.car\_name=clc.car\_name ,

cc.vehicle\_type\_id=clc.vehicle\_type\_id,cc.engine\_type\_id = clc.engine\_type\_id ,

cc.gearbox\_type\_id= clc.gearbox\_type\_id ,cc.model\_id= clc.model\_id ,

cc.repaired\_status\_id=clc.repaired\_status\_id

WHERE DECODE(cc.car\_name,clc.car\_name,0,1)

+DECODE(cc.vehicle\_type\_id,clc.vehicle\_type\_id,0,1)

+DECODE(cc.engine\_type\_id, clc.engine\_type\_id,0,1)+DECODE(cc.gearbox\_type\_id,clc.gearbox\_type\_id,0,1)

+DECODE(cc.model\_id,clc.model\_id,0,1)+DECODE(cc.repaired\_status\_id,clc.repaired\_status\_id,0,1)>0

WHEN NOT MATCHED THEN

INSERT (cc.car\_id, cc.car\_number, cc.car\_name, cc.vehicle\_type\_id, cc.engine\_type\_id,

cc.gearbox\_type\_id, cc.model\_id, cc.repaired\_status\_id, cc.start\_dt, cc.end\_dt)

values (bl\_3nf.seq\_cars.nextval, clc.car\_number, clc.car\_name, clc.vehicle\_type\_id, clc.engine\_type\_id,

clc.gearbox\_type\_id, clc.model\_id, clc.repaired\_status\_id, clc.start\_dt, clc.end\_dt) ;

commit;

EXCEPTION

WHEN OTHERS THEN

RAISE;

END load\_ce\_cars;