|  |
| --- |
|  |
|  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| REVISION HISTORY | | | | | |
| Ver. | Description of Change | Author | Date | Approved | |
| Name | Effective Date |
| 1.0 | Initial status | [Alina](mailto:Kiryl_Bucha@epam.com) Makarets | 03-DEC-2017 |  |  |

Contents

[1. Процесс загрузки данных 3](#_Toc500028308)

[1.1. Таблицы 3](#_Toc500028309)

[1.2. Использование Explicit Cursor 4](#_Toc500028310)

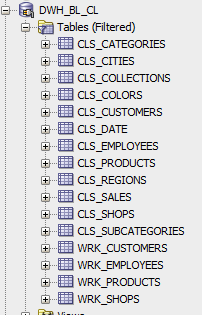
[1.3. Использование implicit курсора 6](#_Toc500028311)

[1.4. Использование MERGE 7](#_Toc500028312)

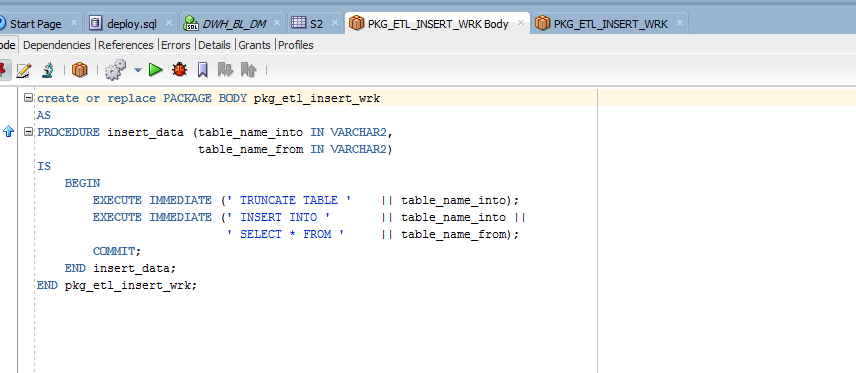
# Процесс загрузки данных

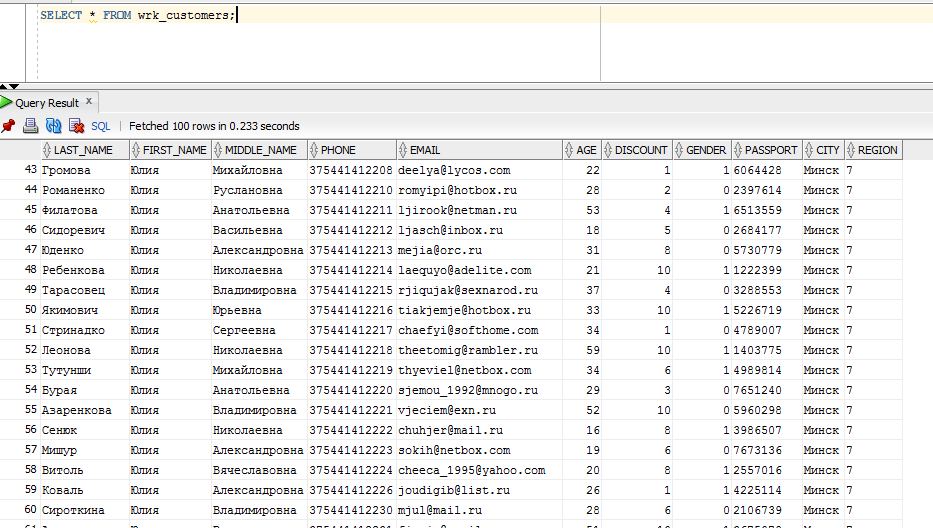
## Таблицы

BL\_CL слой подразумевает собой наличие двух типов таблиц wrk и cls.



Таблиц типов wrk было столько же, сколько и external таблиц, повторяющие полностью их структуру. Для перезаливки данных в данные таблицы был написан один пакет, который использовался для всех таблиц, просто заменяя имя.

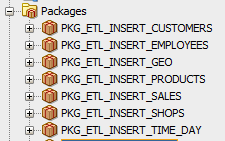


Таблицы типа cls повторяли структуру таблиц, находящихся на следующем слое. При вставке данных в cls таблицы была использована очистка данных:

* Определения пола покупателя и работника
* Сопоставления номера региона и его названию ( 7 – город Минск)
* Определен уникальный ключ для продукта при помощи конкатенации кода продукта и его цвета
* т.д.

Также на данном слое были использован implicit и explicit курсор.

Пакеты, которые были использованы для вставки данных на уровени 3NF.



## Использование Explicit Cursor

Явный курсор - это курсор, который нужно создать явно, а затем управлять им. Нужно объявить и определить явный курсор, указать ему имя и связать его с запросом.

----------------------------------------------------

PROCEDURE insert\_table\_employees

IS

CURSOR emp\_cursor

IS

SELECT TO\_NUMBER(employee\_code) as employee\_id,

first\_name,

last\_name,

middle\_name,

phone,

email,

age,

CASE

WHEN substr (middle\_name,-1) = 'а'

THEN 'Ж'

ELSE 'М'

END AS gender,

passport\_code || passport\_letter as passport,

NVL(to\_date(start\_date, 'DD-MM-YYYY'),'01-ЯНВ-1990') as start\_dt,

NVL(to\_date(end\_date, 'DD-MM-YYYY'),'31-ДЕК-9999') as end\_dt,

CASE

WHEN active = 1

THEN 'Y'

WHEN active = 0

THEN 'N'

END as is\_active

FROM wrk\_employees

WHERE first\_name IS NOT NULL

AND last\_name IS NOT NULL

AND middle\_name IS NOT NULL

AND email IS NOT NULL;

rt\_emp\_cursor cls\_employees%rowtype;

BEGIN

EXECUTE IMMEDIATE 'TRUNCATE TABLE cls\_employees';

OPEN emp\_cursor;

LOOP

FETCH emp\_cursor INTO rt\_emp\_cursor;

IF emp\_cursor%found THEN

INSERT INTO cls\_employees

(

employee\_id,

last\_name,

first\_name,

middle\_name,

phone,

email,

age,

gender,

passport,

start\_dt,

end\_dt,

is\_active

)

VALUES

(

rt\_emp\_cursor.employee\_id,

rt\_emp\_cursor.last\_name,

rt\_emp\_cursor.first\_name,

rt\_emp\_cursor.middle\_name,

rt\_emp\_cursor.phone,

rt\_emp\_cursor.email,

rt\_emp\_cursor.age,

rt\_emp\_cursor.gender,

rt\_emp\_cursor.passport,

rt\_emp\_cursor.start\_dt,

rt\_emp\_cursor.end\_dt,

rt\_emp\_cursor.is\_active

)

;

END IF;

EXIT WHEN emp\_cursor%notfound;

END LOOP;

COMMIT;

CLOSE emp\_cursor;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

RAISE;

END insert\_table\_employees;

## Использование implicit курсора

Примерами неявных курсоров являются INSERT, UPDATE, или SELECT INTOНеявные курсоры не обязательно должны быть объявлены ( DECLARE) , открыты ( OPEN ), использование FETCH. Неявные атрибуты курсора указываются с помощью курсора SQL ( SQL%ROWCOUNT, использующийся для подсчет записей в курсоре).

PROCEDURE insert\_table\_products

IS

BEGIN

DECLARE

CURSOR pro\_cursor IS

SELECT product\_code||'-'||color || '-'|| subcat.subcategory\_surr\_id AS product\_id,

product\_name,

NVL(product\_desc,wrk.subcategory\_name) as product\_desc,

to\_number ( price, '99999999.99') as price,

clr.color\_surr\_id,

col.collection\_surr\_id,

cat.category\_surr\_id,

subcat.subcategory\_surr\_id,

to\_date(wrk.start\_date,'DD-MM-YYYY') as start\_dt

FROM wrk\_products wrk JOIN ce\_colors clr ON wrk.color=clr.color\_id

JOIN ce\_categories cat ON wrk.category\_name=cat.category\_name

JOIN ce\_subcategories subcat ON wrk.subcategory\_name=subcat.subcategory\_name

JOIN ce\_collections col ON wrk.collection\_name=col.collection\_name;

BEGIN

EXECUTE IMMEDIATE (' TRUNCATE TABLE cls\_products');

FOR prod\_cursor\_val IN pro\_cursor LOOP

INSERT INTO cls\_products (

product\_id,

product\_name,

product\_desc,

price,

color\_id,

collection\_id,

category\_id,

subcategory\_id,

start\_dt,

end\_dt,

is\_active

)

VALUES ( prod\_cursor\_val.product\_id,

prod\_cursor\_val.product\_name,

prod\_cursor\_val.product\_desc,

prod\_cursor\_val.price,

prod\_cursor\_val.color\_surr\_id,

prod\_cursor\_val.collection\_surr\_id,

prod\_cursor\_val.category\_surr\_id,

prod\_cursor\_val.subcategory\_surr\_id,

prod\_cursor\_val.start\_dt,

TO\_DATE('31-ДЕК-9999','DD-MON-YYYY'),

'Y'

);

END LOOP;

COMMIT;

END;

EXCEPTION

WHEN OTHERS

THEN RAISE;

END insert\_table\_products;

## Использование MERGE

Практически во всех пакетах был использован merge, но как показала практика данный способ не сильно эффективен на больших объемах данных при использовании UPDATE statement, т.к. данная процедура занимает много времени, но он прекрасно справляется при первой вставке данных в таблицы, находящихся на BL\_3NF слое

PROCEDURE merge\_table\_ce\_colors

IS

BEGIN

MERGE INTO ce\_colors t USING

( SELECT color\_id,

color\_name

FROM cls\_colors

MINUS

SELECT color\_id,

color\_name

FROM ce\_colors

) c

ON ( c.color\_id=t.color\_id )

WHEN matched THEN

UPDATE SET t.color\_name=c.color\_name

WHEN NOT matched THEN

INSERT

(color\_surr\_id,

color\_id,

color\_name,

insert\_dt,

update\_dt)

VALUES

(ce\_colors\_seq.nextval,

c.color\_id,

c.color\_name,

SYSDATE,

SYSDATE) ;

COMMIT ;

EXCEPTION

WHEN OTHERS

THEN RAISE;

END merge\_table\_ce\_colors;