Міністерство освіти і науки України

Національний технічний університет України

“Київський політехнічний інститут”

Кафедра АСОІУ

**ЗВІТ**

про виконання комп’ютерного практикуму № 4

з дисципліни

“ OLAP та сховища даних ”

|  |  |  |
| --- | --- | --- |
| Прийняв: |  | Виконав: |
| Олійник Ю. О. |  | студент 3-го курсу  гр. ІП-51 ФІОТ  Зарічковий Олександр  Анатолійович |

Київ – 2017

**ЗМІСТ:**

[1 Завдання 3](#_Toc514961276)

[2 Виконання Завдання 4](#_Toc514961277)

# Завдання

Завдання 1

Маємо 2 схеми - task1, task3 (КР1, КР3). Створіть пакет з набором процедур/функцій, який виконає завантаження та перетворення даних (INVOICE, SALES, STORE) до відповідних таблиць схеми task3.

# Виконання Завдання

1. Текст процедур
   1. Процедура завантаження даних в stage таблиці, дані на даному етапі очищуються для подальшої обробки.

*Текст запиту*:

CREATE OR REPLACE PROCEDURE LOAD\_INVOICE IS

CURSOR C\_INVOICE IS

SELECT DISTINCT

ID\_STUFF,

STAFF\_NAME,

E\_MAIL,

INVOICE,

SUPPLIER,

PRODUCT,

QUANTITY,

PRICE,

INVOICE\_DATE

FROM INVOICE

WHERE REGEXP\_LIKE (E\_MAIL, '^[A-ZA-Z]+[A-ZA-Z0-9. ]+@[A-ZA-Z0-9.-]+\.[A-ZA-Z]{2,4}$')

AND REGEXP\_LIKE (SUPPLIER, '^[A-ZA-Z]')

AND REGEXP\_LIKE (PRODUCT, '^TV-+[0-9]+$')

AND REGEXP\_LIKE (INVOICE, '^INV-([0-9][0-9]|[0-9][0-9][0-9])$')

AND REGEXP\_LIKE (QUANTITY, '^+[0-9]+$')

AND REGEXP\_LIKE (INVOICE\_DATE, '^(0[1-9]|[12][0-9]|3[01])[- /.](0[1-9]|1[012])[- /.](19|20)\d\d$');

BEGIN

FOR INVOICE\_ROW IN C\_INVOICE

LOOP

INSERT INTO S\_INVOICE (ID\_STUFF, NAME, SURNAME, EMAIL, INVOICE,

SUPPLIER, PRODUCT, QUANTITY, PRICE, INVOICE\_DATE)

VALUES (INVOICE\_ROW.ID\_STUFF, REGEXP\_SUBSTR (INVOICE\_ROW.STAFF\_NAME, '[^ ]+', 1, 1),

REGEXP\_SUBSTR (INVOICE\_ROW.STAFF\_NAME, '[^ ]+', 1, 2), INVOICE\_ROW.E\_MAIL,

REGEXP\_SUBSTR (INVOICE\_ROW.INVOICE, '[^-]+', 1, 2), INVOICE\_ROW.SUPPLIER,

INVOICE\_ROW.PRODUCT, INVOICE\_ROW.QUANTITY, INVOICE\_ROW.PRICE, INVOICE\_ROW.INVOICE\_DATE);

END LOOP;

END;

CREATE OR REPLACE PROCEDURE LOAD\_STORE IS

CURSOR C\_STORE IS

SELECT DISTINCT

REGEXP\_SUBSTR (STUFF\_NAME, '[^ ]+', 1, 1) AS NAME,

REGEXP\_SUBSTR (STUFF\_NAME, '[^ ]+', 1, 2) AS SURNAME,

SUPPLIER,

SHELF,

PRODUCT,

QUANTITY,

OPER\_TYPE,

STORE\_DATE

FROM STORE

WHERE SHELF < 100

AND REGEXP\_LIKE (STORE\_DATE, '^(0[1-9]|[12][0-9]|3[01])[- /.](0[1-9]|1[012])[- /.](19|20)\d\d$')

AND REGEXP\_LIKE(OPER\_TYPE, '^+(IN|OUT)+$', 'i')

AND REGEXP\_LIKE (QUANTITY, '^+[0-9]+$')

AND REGEXP\_LIKE (SUPPLIER, '^[a-zA-Z]+$')

AND REGEXP\_LIKE (PRODUCT, '^TV-+[0-9]+$')

AND REGEXP\_LIKE (REGEXP\_SUBSTR (STUFF\_NAME, '[^ ]+', 1, 1), '^[a-zA-Z]+$')

AND REGEXP\_LIKE (REGEXP\_SUBSTR (STUFF\_NAME, '[^ ]+', 1, 2), '^[a-zA-Z]+$');

BEGIN

FOR STORE\_ROW IN C\_STORE

LOOP

INSERT INTO S\_STORE (NAME, SURNAME, SUPPLIER, SHELF, PRODUCT,

QUANTITY, OPER\_TYPE, STORE\_DATE)

VALUES (STORE\_ROW.NAME, STORE\_ROW.SURNAME, STORE\_ROW.SUPPLIER,

STORE\_ROW.SHELF, STORE\_ROW.PRODUCT, STORE\_ROW.QUANTITY, STORE\_ROW.OPER\_TYPE, STORE\_ROW.STORE\_DATE);

END LOOP;

END;

* 1. Створення sequence для генерування ключів.

*Текст запиту*:

CREATE SEQUENCE STUFF\_ID

START WITH 1

INCREMENT BY 1

NOCACHE

NOCYCLE;

CREATE SEQUENCE SUPPLIER\_ID

START WITH 1

INCREMENT BY 1

NOCACHE

NOCYCLE;

CREATE SEQUENCE PRODUCT\_ID

START WITH 1

INCREMENT BY 1

NOCACHE

NOCYCLE;

CREATE SEQUENCE OPER\_TYPE\_ID

START WITH 1

INCREMENT BY 1

NOCACHE

NOCYCLE;

CREATE SEQUENCE STORE\_ID

START WITH 1

INCREMENT BY 1

NOCACHE

NOCYCLE;

* 1. Запис даних за допомогою merge в кінцеві таблиці.

*Текст запиту*:

MERGE INTO D\_STUFF D USING

(

SELECT DISTINCT

ID\_STUFF,

NAME,

SURNAME,

EMAIL

FROM S\_INVOICE

)

S ON (S.ID\_STUFF = D.ID\_STUFF)

WHEN MATCHED THEN

UPDATE SET NAME = S.NAME, SURNAME = S.SURNAME, EMAIL = S.EMAIL

WHEN NOT MATCHED THEN

INSERT VALUES (S.ID\_STUFF, 1, S.NAME, S.SURNAME, NULL,

NULL, S.EMAIL);

MERGE INTO D\_STUFF D USING

(

SELECT DISTINCT

NAME,

SURNAME

FROM S\_STORE

)

S ON (S.NAME = D.NAME AND S.SURNAME = D.SURNAME)

WHEN NOT MATCHED THEN

INSERT VALUES (STUFF\_ID.NEXTVAL, 1, S.NAME, S.SURNAME, NULL,

NULL, NULL);

MERGE INTO D\_SUPPLIER D USING

(

SELECT DISTINCT

SUPPLIER

FROM S\_INVOICE

)

S ON (S.SUPPLIER = D.SUPPLIER\_NAME)

WHEN NOT MATCHED THEN

INSERT VALUES (SUPPLIER\_ID.NEXTVAL, S.SUPPLIER, NULL);

MERGE INTO D\_SUPPLIER D USING

(

SELECT DISTINCT

SUPPLIER

FROM S\_STORE

)

S ON (S.SUPPLIER = D.SUPPLIER\_NAME)

WHEN NOT MATCHED THEN

INSERT VALUES (SUPPLIER\_ID.NEXTVAL, S.SUPPLIER, NULL);

MERGE INTO D\_PRODUCT D USING

(

SELECT DISTINCT

T1.PRODUCT,

T1.PRICE,

T2.ID\_SUPPLIER

FROM S\_INVOICE T1

JOIN D\_SUPPLIER T2

ON (T1.SUPPLIER = T2.SUPPLIER\_NAME)

)

S ON (S.PRODUCT = D.PRODUCT\_NAME AND D.ID\_SUPPLIER = S.ID\_SUPPLIER)

WHEN MATCHED THEN

UPDATE SET

PRICE = S.PRICE

WHEN NOT MATCHED THEN

INSERT VALUES (PRODUCT\_ID.NEXTVAL, 1, S.ID\_SUPPLIER, 1,

S.PRODUCT, NULL, S.PRICE);

MERGE INTO D\_PRODUCT D USING

(

SELECT DISTINCT

T1.PRODUCT,

T2.ID\_SUPPLIER

FROM S\_STORE T1

JOIN D\_SUPPLIER T2

ON (T1.SUPPLIER = T2.SUPPLIER\_NAME)

)

S ON (S.PRODUCT = D.PRODUCT\_NAME AND D.ID\_SUPPLIER = S.ID\_SUPPLIER)

WHEN NOT MATCHED THEN

INSERT VALUES (PRODUCT\_ID.NEXTVAL, 1, S.ID\_SUPPLIER, 1,

S.PRODUCT, NULL, 1000);

MERGE INTO D\_INVOICE D USING

(

SELECT DISTINCT

T1.INVOICE,

T1.ID\_STUFF,

T1.INVOICE\_DATE

FROM S\_INVOICE T1

)

S ON (D.ID\_INVOICE = S.INVOICE)

WHEN MATCHED THEN

UPDATE SET ID\_STUFF = S.ID\_STUFF, PURCHASE\_TIME = S.INVOICE\_DATE

WHEN NOT MATCHED THEN

INSERT VALUES (S.INVOICE, 1, S.ID\_STUFF, S.INVOICE\_DATE);

MERGE INTO D\_INVOICE\_DETAIL D USING

(

SELECT DISTINCT

T1.INVOICE,

T3.ID\_PRODUCT,

T1.QUANTITY

FROM S\_INVOICE T1

JOIN D\_SUPPLIER T2

ON (T1.SUPPLIER = T2.SUPPLIER\_NAME)

JOIN D\_PRODUCT T3

ON (T1.PRODUCT = T3.PRODUCT\_NAME AND T2.ID\_SUPPLIER = T3.ID\_SUPPLIER)

)

S ON (D.ID\_INVOICE = S.INVOICE AND D.ID\_PRODUCT = S.ID\_PRODUCT)

WHEN MATCHED THEN

UPDATE SET QUANTITY = S.QUANTITY

WHEN NOT MATCHED THEN

INSERT VALUES (S.INVOICE, S.ID\_PRODUCT, S.QUANTITY, NULL);

MERGE INTO D\_TYPE\_OPER D USING

(

SELECT DISTINCT

T1.OPER\_TYPE

FROM S\_STORE T1

)

S ON (UPPER(D.NAME\_OPER) = UPPER(S.OPER\_TYPE))

WHEN NOT MATCHED THEN

INSERT VALUES (OPER\_TYPE\_ID.NEXTVAL, UPPER(S.OPER\_TYPE), NULL);

MERGE INTO D\_STORE D USING

(

SELECT DISTINCT

T2.ID\_PRODUCT,

T1.SHELF,

T1.STORE\_DATE,

T3.ID\_OPER\_TYPE,

T1.QUANTITY

FROM S\_STORE T1

JOIN D\_SUPPLIER T4

ON (T1.SUPPLIER = T4.SUPPLIER\_NAME)

JOIN D\_PRODUCT T2

ON (T1.PRODUCT = T2.PRODUCT\_NAME AND T2.ID\_SUPPLIER = T4.ID\_SUPPLIER)

JOIN D\_TYPE\_OPER T3

ON (T3.NAME\_OPER = UPPER(T1.OPER\_TYPE))

)

S ON (D.ID\_PRODUCT = S.ID\_PRODUCT AND D.SHELF = S.SHELF AND D.DATE\_OPER = S.STORE\_DATE

AND D.ID\_OPER\_TYPE = S.ID\_OPER\_TYPE)

WHEN MATCHED THEN

UPDATE SET QUANTITY = S.QUANTITY

WHEN NOT MATCHED THEN

INSERT VALUES (STORE\_ID.NEXTVAL, S.ID\_PRODUCT, S.SHELF, S.STORE\_DATE, S.ID\_OPER\_TYPE,

S.QUANTITY);