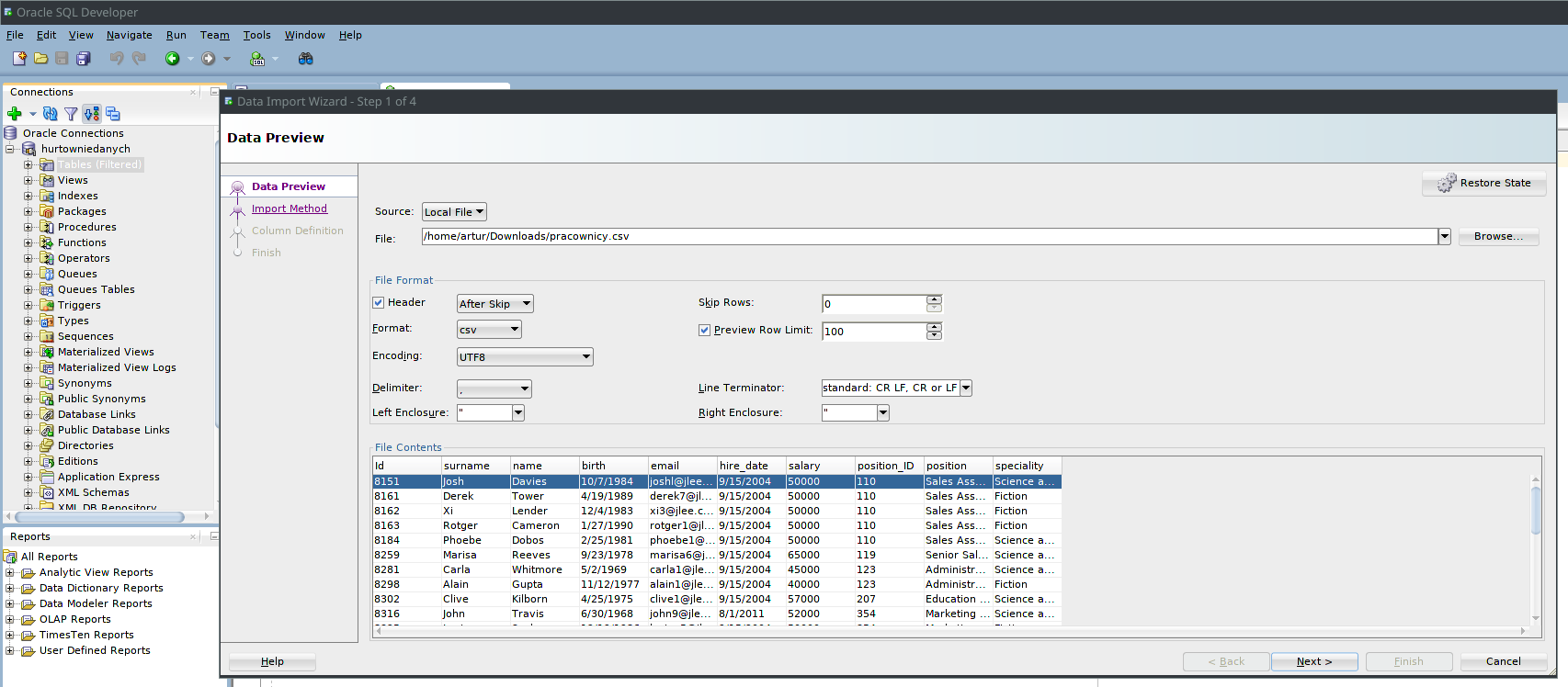
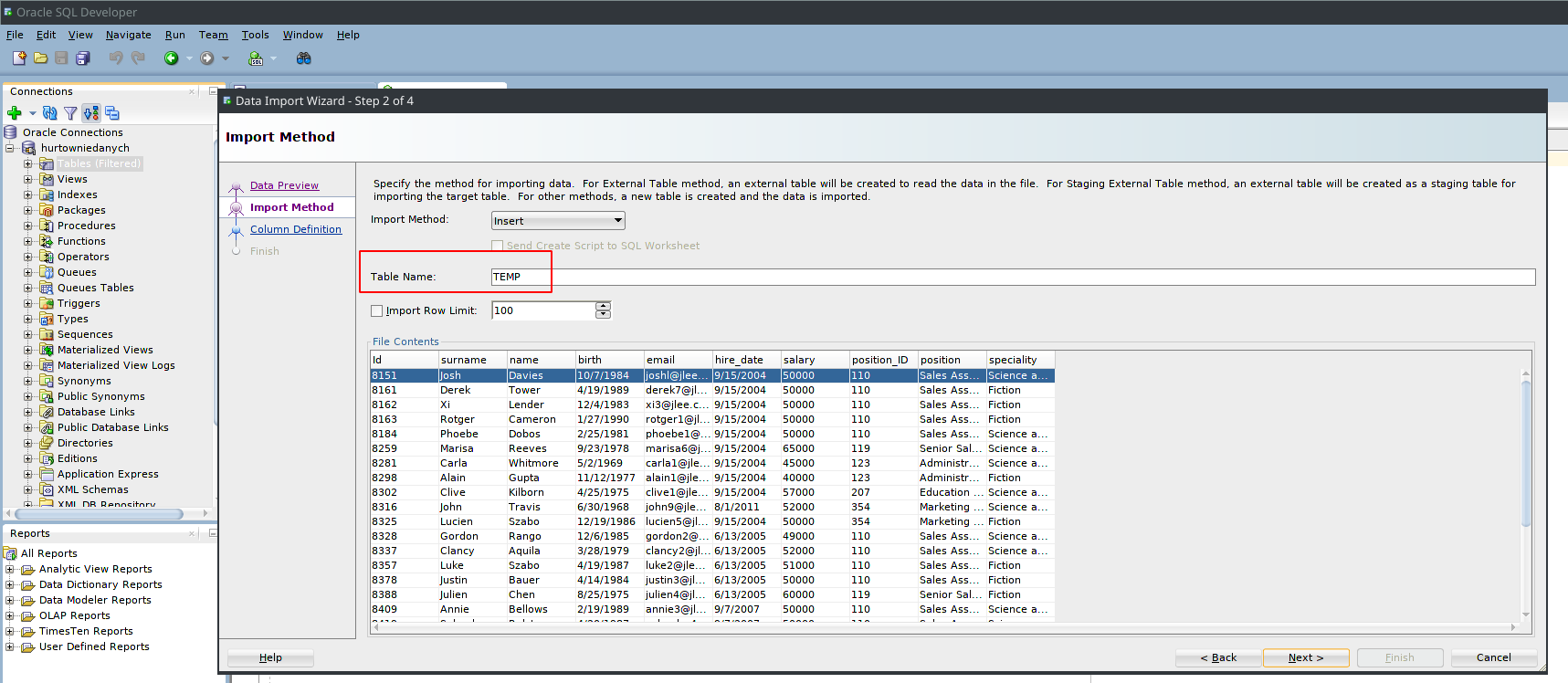
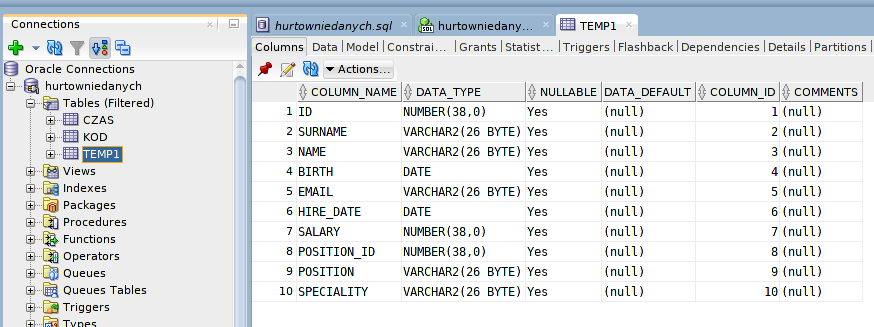
# Zadanie 1 – załadowanie danych z pliku CSV

Zrzuty ekranu prezentujące proces importowania danych przy użyciu narzędzia do tego celu w Oracle SQL Developer:

  
Figure 1: Wybieranie poprawnych typów danych dla zaimportowanych danych

  
Figure 2: Tworzenie tabeli TEMP1 (nazwa TEMP była zajęta)

  
Figure 3: Typy danych stworzonej tabeli TEMP1 (na podstawie plików z pracownicy.csv)

# Zadanie 2 – stworzenie tabel

Kod SQL tworzący tabele Stanowisko, Specjalność oraz Pracownik:

CREATE TABLE stanowisko (

id\_stanowiska NUMBER PRIMARY KEY,

nazwa VARCHAR(50)

);

CREATE TABLE specjalnosc (

id\_specjalnosci NUMBER PRIMARY KEY,

nazwa VARCHAR(50)

);

CREATE TABLE pracownik (

id\_pracownika NUMBER PRIMARY KEY,

surname VARCHAR(50),

name VARCHAR(50),

birth DATE NOT NULL,

hire DATE NOT NULL,

email VARCHAR(50),

salary NUMBER NOT NULL,

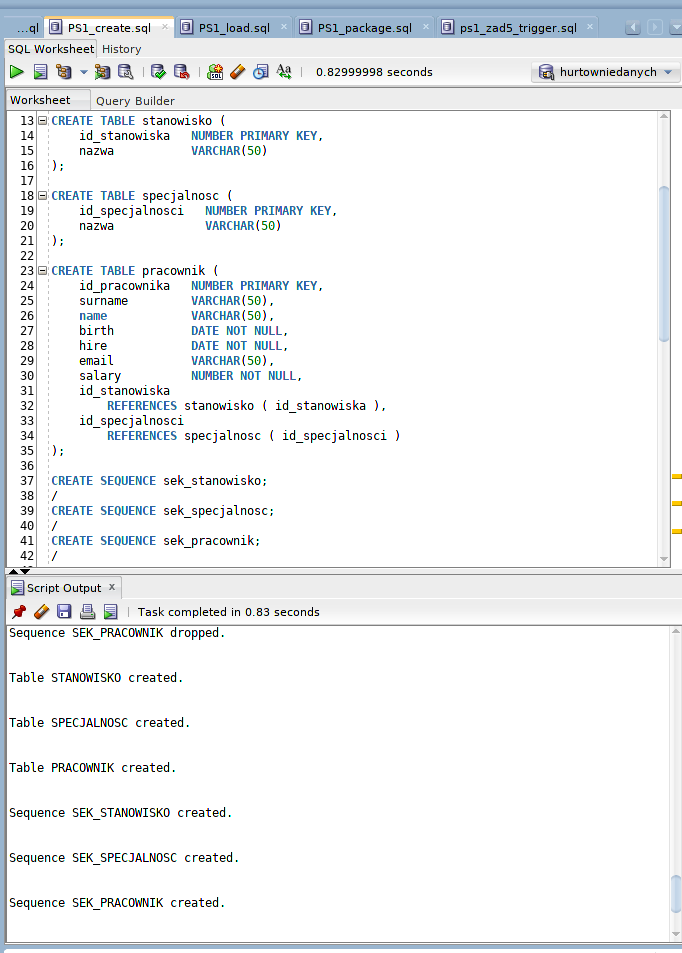
id\_stanowiska

REFERENCES stanowisko ( id\_stanowiska ),

id\_specjalnosci

REFERENCES specjalnosc ( id\_specjalnosci )

);

  
Figure 4: Wynik po uruchomieniu kodu tworzącego tabele Stanowisko, Specjalność i Pracownik

# Zadanie 3 – sekwencje

Kod SQL tworzący sekwencje do auto-inkrementacji klucza głównego tabel Stanowisko, Specjalnosc I Pracownik:

CREATE SEQUENCE sek\_stanowisko;

/

CREATE SEQUENCE sek\_specjalnosc;

/

CREATE SEQUENCE sek\_pracownik;

/

CREATE OR REPLACE TRIGGER t\_ustaw\_id\_stanowisko BEFORE

INSERT ON stanowisko

FOR EACH ROW

BEGIN

:new.id\_stanowiska := sek\_stanowisko.nextval;

END;

/

CREATE OR REPLACE TRIGGER t\_ustaw\_id\_specjalnosc BEFORE

INSERT ON specjalnosc

FOR EACH ROW

BEGIN

:new.id\_specjalnosci := sek\_specjalnosc.nextval;

END;

/

CREATE OR REPLACE TRIGGER t\_ustaw\_id\_pracownik BEFORE

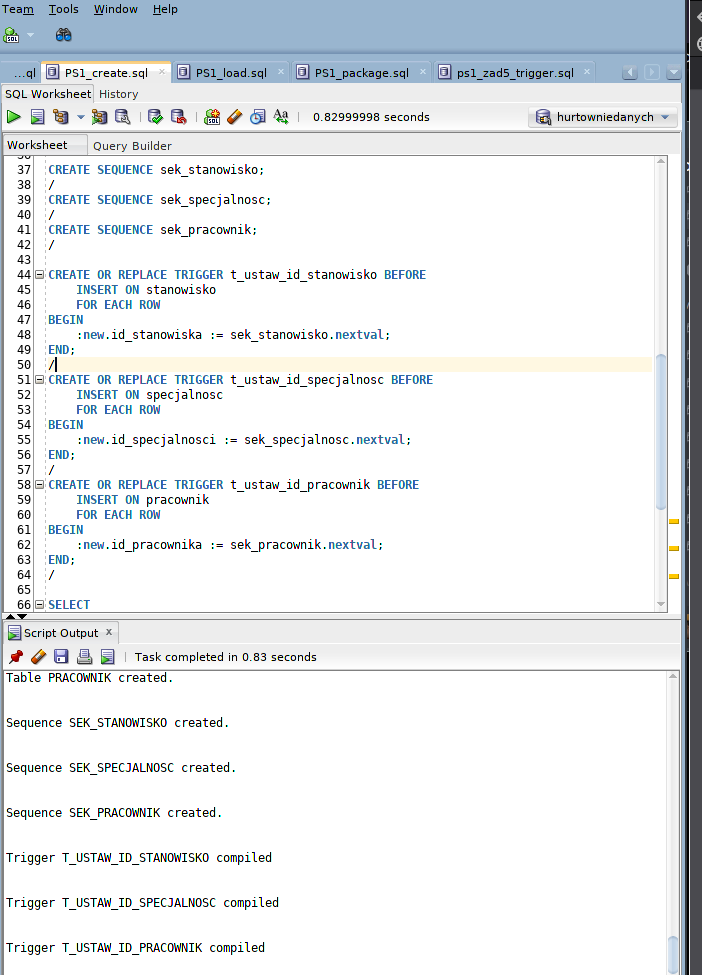
INSERT ON pracownik

FOR EACH ROW

BEGIN

:new.id\_pracownika := sek\_pracownik.nextval;

END;

  
Figure 5: Wynik wykonania kodu tworzacego sekwencje do auto-inkrementacji klucza głównego

# Zadanie 4 – pakiet (średnia pensja I ładowanie)

Kod SQL tworzący pakiet oraz ciało pakietu. Pakiet zawiera procedurę “reshape\_data”, która pobiera zawartość tablicy TEMP1 po czym przekształca wiersze w taki sposób, aby uzupełnić tablice Pracownik, Specjalność oraz Stanowisko.

CREATE OR REPLACE PACKAGE zad4oraz5 AUTHID definer AS

PROCEDURE reshape\_data;

FUNCTION avg\_salary (

nazwa\_stanowiska VARCHAR2

) RETURN NUMBER;

END zad4oraz5;

/

CREATE OR REPLACE PACKAGE BODY zad4oraz5 AS

PROCEDURE reshape\_data IS

BEGIN

INSERT INTO specjalnosc ( nazwa )

SELECT DISTINCT

speciality AS name

FROM

temp1;

INSERT INTO stanowisko ( nazwa )

SELECT DISTINCT

position AS nazwa

FROM

temp1;

INSERT INTO pracownik (

surname,

name,

birth,

email,

hire,

salary,

id\_stanowiska,

id\_specjalnosci

)

SELECT

temp1.surname,

temp1.name,

temp1.birth,

temp1.email,

temp1.hire\_date,

temp1.salary,

stanowisko.id\_stanowiska,

specjalnosc.id\_specjalnosci

FROM

temp1

INNER JOIN stanowisko ON temp1.position = stanowisko.nazwa

INNER JOIN specjalnosc ON temp1.speciality = specjalnosc.nazwa;

END reshape\_data;

FUNCTION avg\_salary (

nazwa\_stanowiska VARCHAR2

) RETURN NUMBER IS

srednia\_pensja NUMBER;

BEGIN

SELECT

avgsal

INTO srednia\_pensja

FROM

(

SELECT

AVG(pracownik.salary) AS avgsal,

pracownik.id\_stanowiska AS idst

FROM

pracownik

GROUP BY

pracownik.id\_stanowiska

)

INNER JOIN stanowisko ON stanowisko.id\_stanowiska = idst

WHERE

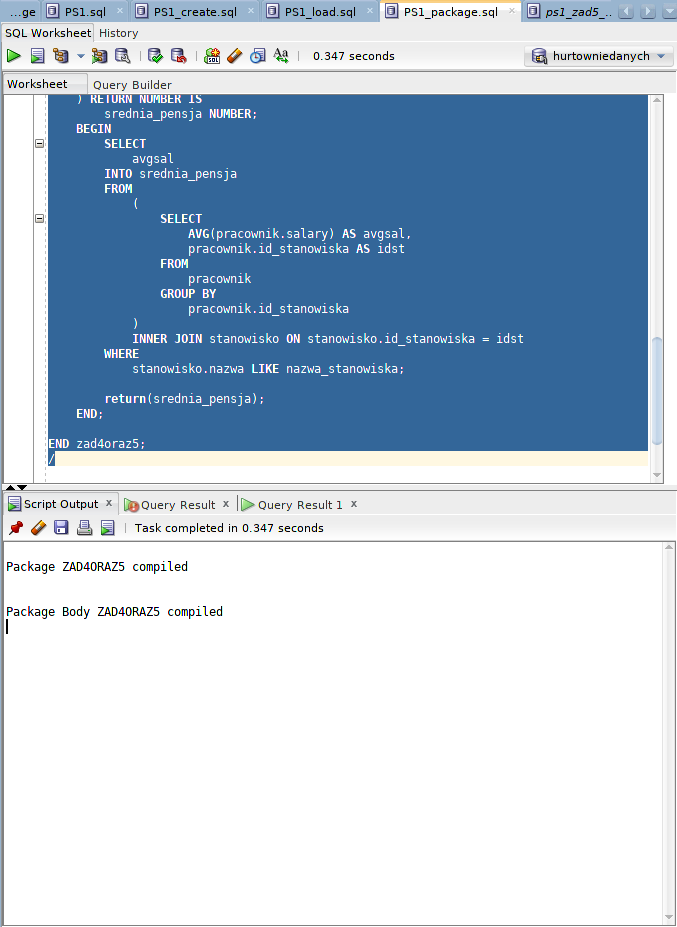
stanowisko.nazwa LIKE nazwa\_stanowiska;

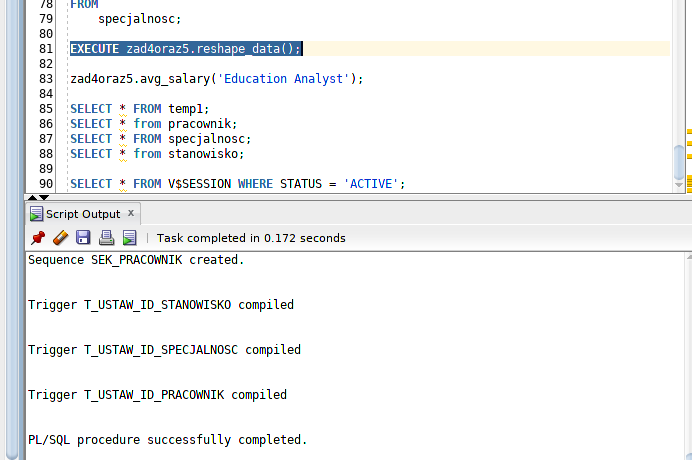
**return**(srednia\_pensja);

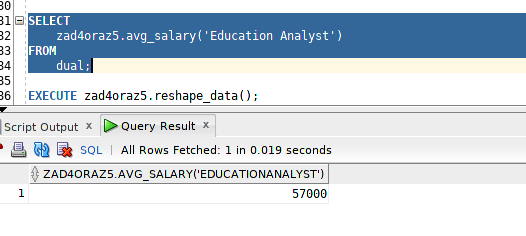
END;

END zad4oraz5;

/

  
Figure 6: Wynik stworzenia pakietu oraz ciała (interfejsu) pakietu

  
Figure 7: Wynik wywołania procedury uzupełniającej tabele Stanowisko, Pracownik i Specjalność na podstawie tabeli TEMP1 (zawierającej dane z pliku pracownicy.csv). Procedura nazywa się “reshape\_data”

  
Figure 8: Wynik wywołania funkcji "avg\_salary", która zwraca średnią pensję na danym stanowisku. Nazwa stanowiska jest podawana do funkcji poprzez parametr. W tym wypadku badanym stanowiskiem jest 'Education Analyst', którego średnia pensja wynosi 57000

# Zadanie 5 – wyzwalacze

Kod SQL tworzacy wyzwalacze, które zabraniają dodania (lub zmianę - UPDATE) użytkownika z pensją większą niż 30% od średniej (poniżej). Wyzwalacze używają wewnątrz funkcji “avg\_salary” z poprzedniego zadania.

CREATE OR REPLACE TRIGGER ogranicz\_pensje\_insert BEFORE

INSERT ON pracownik

FOR EACH ROW

DECLARE

avg\_salary\_result NUMBER;

BEGIN

SELECT

zad4oraz5.avg\_salary(st.nazwa)

INTO avg\_salary\_result

FROM

stanowisko st

WHERE

st.id\_stanowiska = :new.id\_stanowiska;

IF ( :new.salary > 1.3 \* avg\_salary\_result ) THEN

raise\_application\_error(-20001, ***'Pensja nie może być większa o 30% od średniej pensji na danym stanowisku'***);

END IF;

END;

/

CREATE OR REPLACE TRIGGER ogranicz\_pensje\_update BEFORE

UPDATE ON pracownik

FOR EACH ROW

DECLARE

avg\_salary\_result NUMBER;

BEGIN

SELECT

zad4oraz5.avg\_salary(st.nazwa)

INTO avg\_salary\_result

FROM

stanowisko st

WHERE

st.id\_stanowiska = :new.id\_stanowiska;

IF ( :new.salary > 1.3 \* avg\_salary\_result ) THEN

raise\_application\_error(-20001, ***'Pensja nie może być większa o 30% od średniej pensji na danym stanowisku'***);

END IF;

END;

INSERT INTO tagisow\_artur.pracownik (

surname,

name,

birth,

hire,

email,

salary,

id\_stanowiska,

id\_specjalnosci

) VALUES (

***'Julien'***,

***'Chen '***,

TO\_DATE(***'25-AUG-75'***, ***'DD-MON-RR'***),

TO\_DATE(***'13-JUN-05'***, ***'DD-MON-RR'***),

***'julien4@jlee.com'***,

999999,

21,

21

);

/

UPDATE pracownik

SET

surname = ***'Julien'***,

name = 'Chen',

birth = TO\_DATE(***'25-AUG-75'***, ***'DD-MON-RR'***),

hire = TO\_DATE(***'13-JUN-05'***, ***'DD-MON-RR'***),

email = ***'julien4@jlee.com'***,

salary = 999999,

id\_stanowiska = 21,

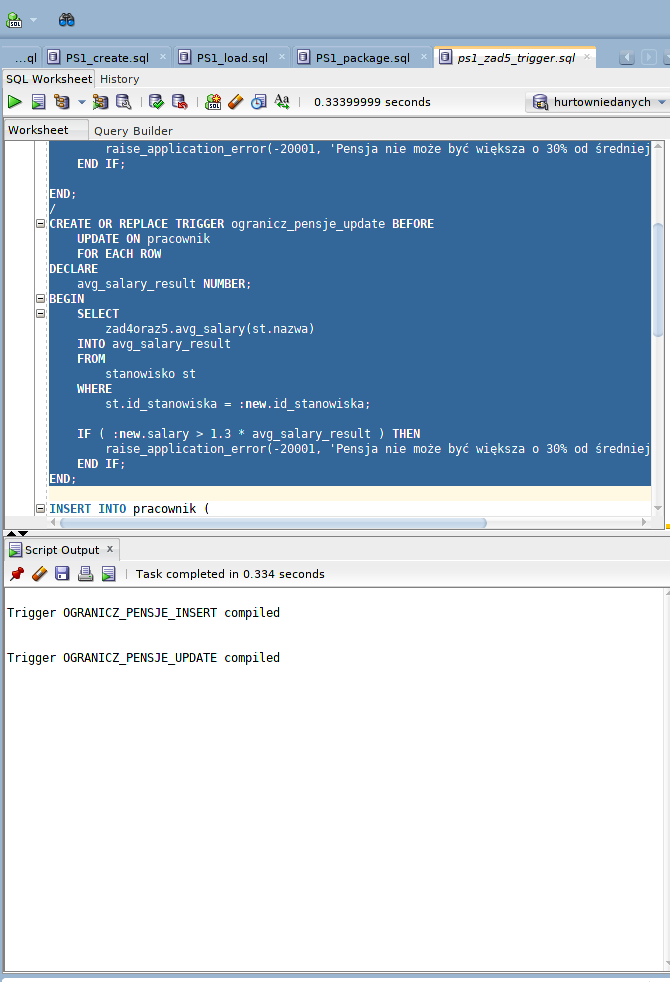
id\_specjalnosci = 21

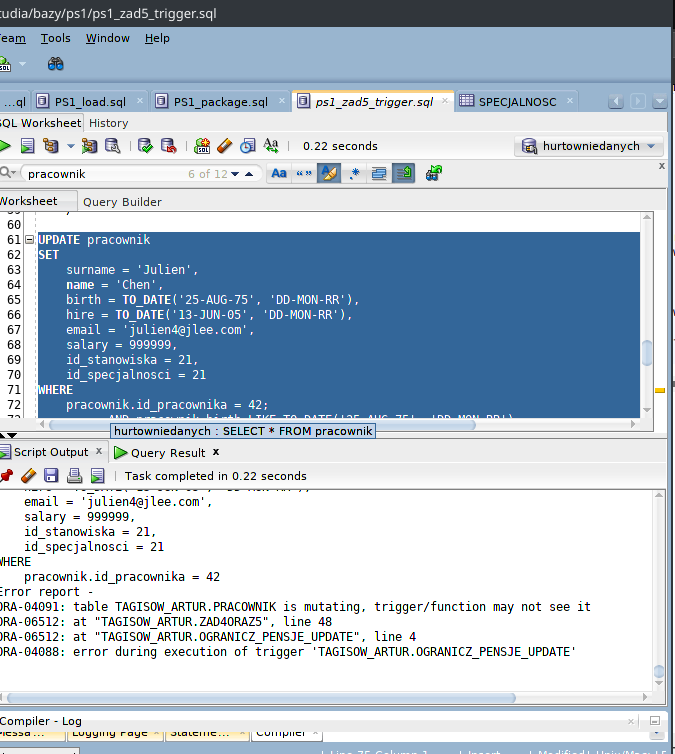
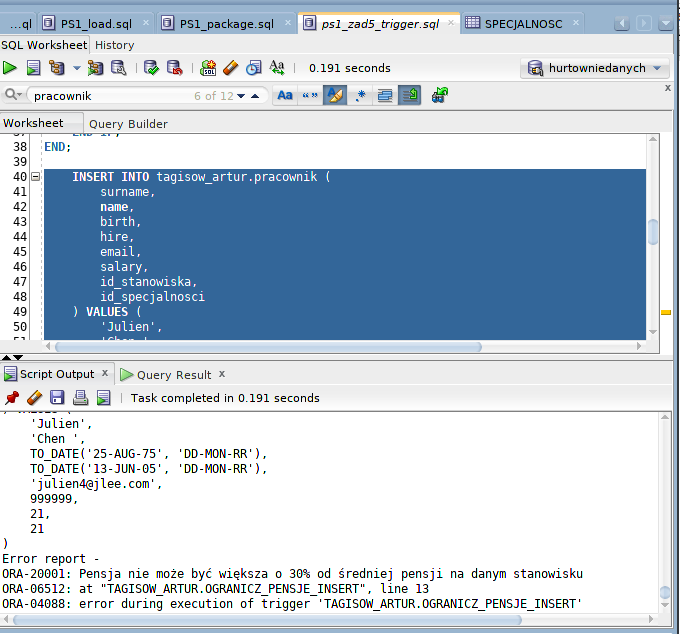
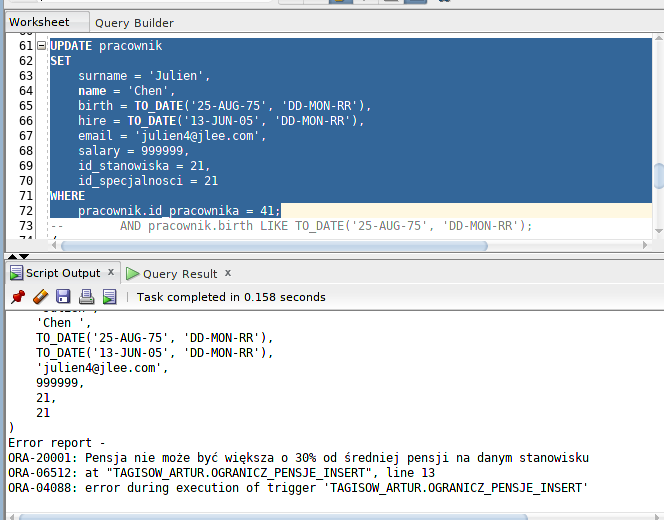
WHERE

pracownik.id\_pracownika = 41;

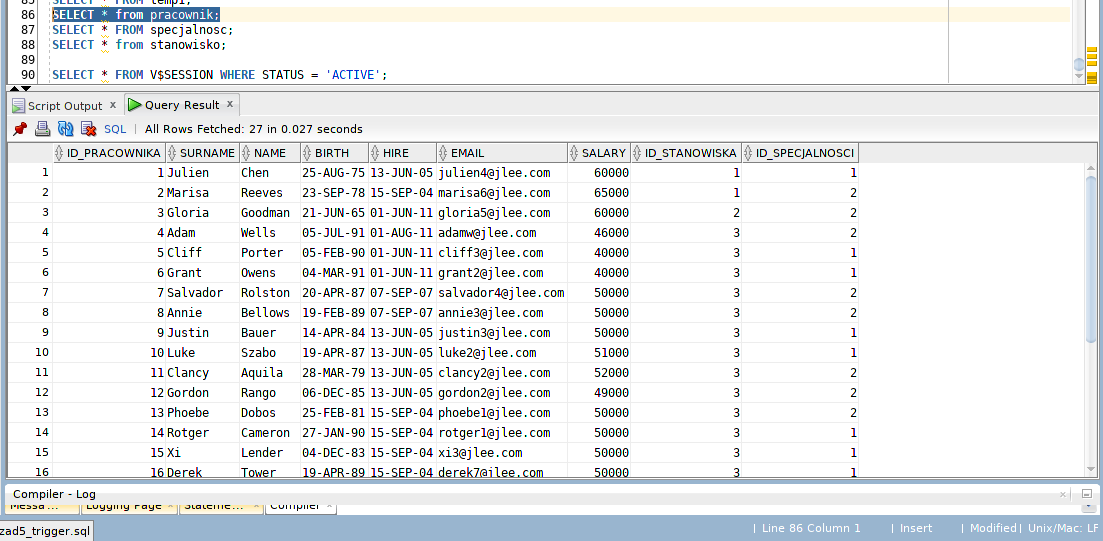
-- AND pracownik.birth LIKE TO\_DATE(***'25-AUG-75'***, ***'DD-MON-RR'***);

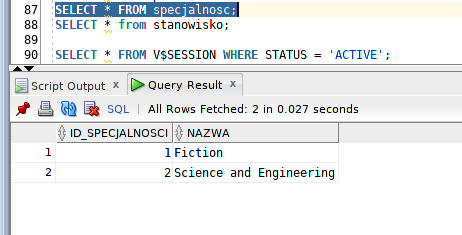
/

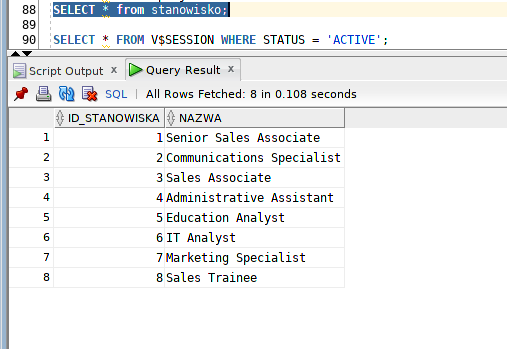




Zrzuty ekranu tabel pracownik, specjalizacja, stanowisko (id mogą nieco się różnić w niektórych screenach – czasami id specjalności mogły mieć id 21 I 22.

  
Figure 9: Zrzut ekranu tabeli Pracownik

  
Figure 10: Zrzut ekranu tabeli Specjalnosc

  
Figure 11: Zrzut ekranu tabeli Stanowisko

# Wnioski

Realizacja wszystkich zadań powiodła się z wyjątkiem:

* Zadania 5 – wyzwalacz “before update” - z nieznanego powodu wywołanie polecenia UPDATE na tabei po ząłożeni tego wyzwalacza zawsze otrzymano błąd o tym, że tabela PRACOWNIK jest aktualnie modyfikowana, niezależnie od momentu w którym momencie zostało ono wykonane (np. Po godzinie od ostatniej operacji CREATE, INSERT itp.)
* Zadania 3 – skryptu ładującego przy użyciu MERGE

Oracle SQL Developer jest dość mało stabilnym programem na systemie operacyjnym Linux. W niektórych przypadkach może to spowodować stworzenie niezamkniętej sesji blokującej dostęp do pewnej tabeli. Praca na uczelnianym serwerze Oracle jest o tyle trudniejsza, że nie istnieje możliwość ręcznego zamknięcia takiej sesji. W takich przypadkach potrzebna jest pomoc administratora.

Warto dodać, że z nieznanego powodu, najmniejszym ID pracownika było np. 21, kiedy oczekiwaną pierwszą wartością jest oczywiście 1. Problem pojawiał się oraz znikał podczas kolejnych importów danych z tabeli TEMP1. Nie jest to szkodliwe zjawisko. Jednak jest na pewno niespodziewane.