

RĪGAS TEHNISKĀ UNIVERSITĀTE

Datorzinātnes un informācijas tehnoloģijas fakultāte

1.praktiskais darbs

**Poligona un testa tabulu izveide un parbaude**

Izstrādāja: Natans Šalamberidze

171RMC203

Pārbaudīja: lektors Jānis Eiduks

2023./2024. māc. Gads

**1.** Datu bāzes sistēmas Oracle21c distribūt-failu nokopēšana (download) no firmas Oracle  web mājas lapas un sistēmas intalēšana (mācību  materiālos ir dots piemērs).  
**2.** Lai varētu izpildīt nepieciešamos SQL vaicājumus un PL/SQL programmas ir nepieciešams redaktors, kas savienojas (pieslēdzas) datu bāzes sistēmai un ļauj ievadīt vēlamos pieprasījumus. Datu bāzes sistēmai Oracle ir iebūvēts komandrindas redaktors SQL Plus, bet viekāršāk un uzskatāmāk ir izmantot redaktoru SQL Developer, kas izstrādāts speciāli datu bāzes sistēmai Oracle. Jāveic redaktora SQL Developer failu kopēšana  no atbilstšās web lapas un jāizpilda tā instalēšana (praktiski arhīva faila atvēršana).

**3.** Izmantojot redaktoru SQL Developer jāpieslēdzas datu bāzes sistēmai Oracle21c (jāievada savienojuma informācija). Sākumā jāizmanto iebūvētais lietotājs SYSTEM ar paroli pēc noklusējuma vai instalējot nomainīto paroli.

**4.**  Redaktorā SQL Developer jādefinē neliela tabula (CREATE TABLE), jāievada daži tabulas raksti (INSERT) un jāveic daži datu pieprasījumi (SELECT), lai pārliecinātos, ka darba poligons strādā.

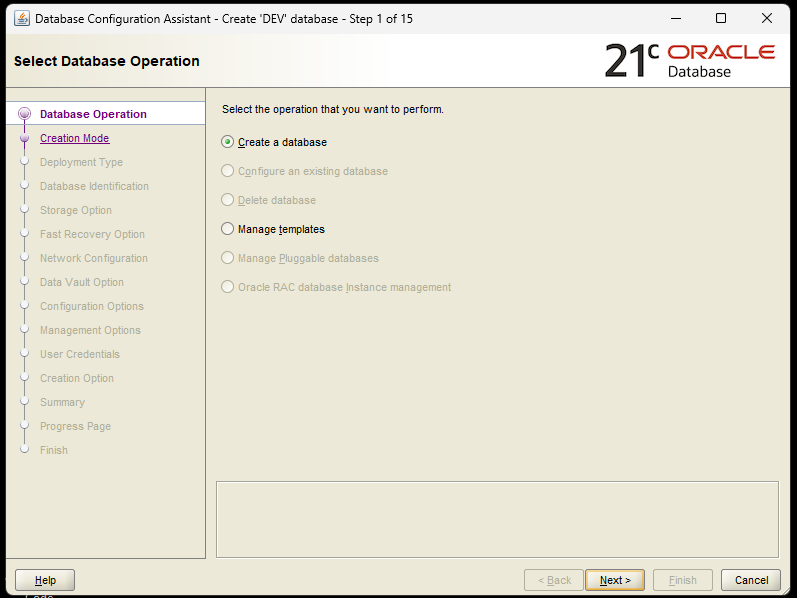
**5.** Turpinājumā jādefinē zvaignes struktūras tabulas. Struktūru veido datu pamattabula un vismaz 3 piesaistes tabulas, kuras tiek izmantotas, lai varētu norādīt vaicājumu meklēšanas noteikumus. Piemēram,  studentu atzīmes - pamattabula. Studentu saraksts, mācību gadiun semestri, universitātes un fakultātes, mācību priekšmeti un pasniedzēji - meklēšanas noteikumu tabulas. Šī struktūra tiks izmantota nākošajos praktiskajos darbos.   
**6.** Pamattabulai rindu skaits ir >100 000. Tabulu aizpildīšanai ar datiem jāuzraksta speciāla PL/SQL programma, kurā datu ģenerēšanai tiek izmantoti gadījuma vērtību ģeneratori DBMS\_RANDOM. VALUE() (gadījuma skaitļi) un DBMS\_RANDOM.STRING() (gadījuma simbolu rindas). Tabulu aizpildi jāveic ar INSERT vai UPDATE komandām.   
**7.** Tabulu pārbaudei jāuzraksta un jāizpilda 3 SQL vaicājumi pamattabulas datu agregātvērtību SUM(), COUNT(), AVG() iegūšanai. Meklēšanas noteikumi tiek norādīti izmantojot pakārtotās tabulas.

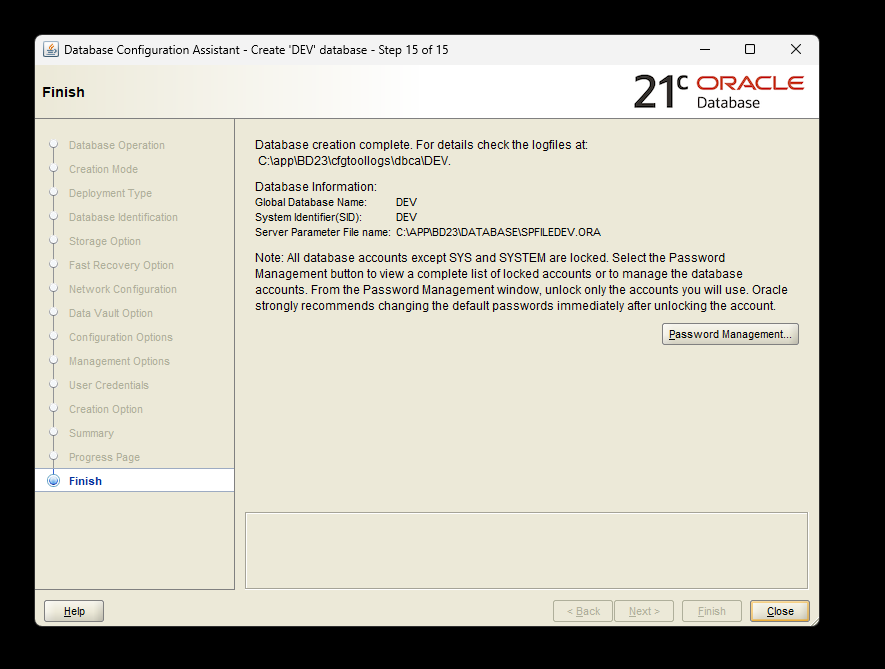
8 Vaicājumu izpildes laiku noteikšanai jāizmanto komanda EXPLAIN PLAN vai funkcijas DBMS\_

9  Izpildītā darba izvērtēšana un secinājumu uzrakstīšana.

**1. Datu bāzes sistēmas Oracle21c distribūt-failu nokopēšana (download) no firmas Oracle  web mājas lapas un sistēmas intalēšana (mācību  materiālos ir dots piemērs).  
2. Lai varētu izpildīt nepieciešamos SQL vaicājumus un PL/SQL programmas ir nepieciešams redaktors, kas savienojas (pieslēdzas) datu bāzes sistēmai un ļauj ievadīt vēlamos pieprasījumus. Datu bāzes sistēmai Oracle ir iebūvēts komandrindas redaktors SQL Plus, bet viekāršāk un uzskatāmāk ir izmantot redaktoru SQL Developer, kas izstrādāts speciāli datu bāzes sistēmai Oracle. Jāveic redaktora SQL Developer failu kopēšana  no atbilstšās web lapas un jāizpilda tā instalēšana (praktiski arhīva faila atvēršana).**

Ir redzama detalizēta datu bāzes instalēšana (Oracle21c versija)

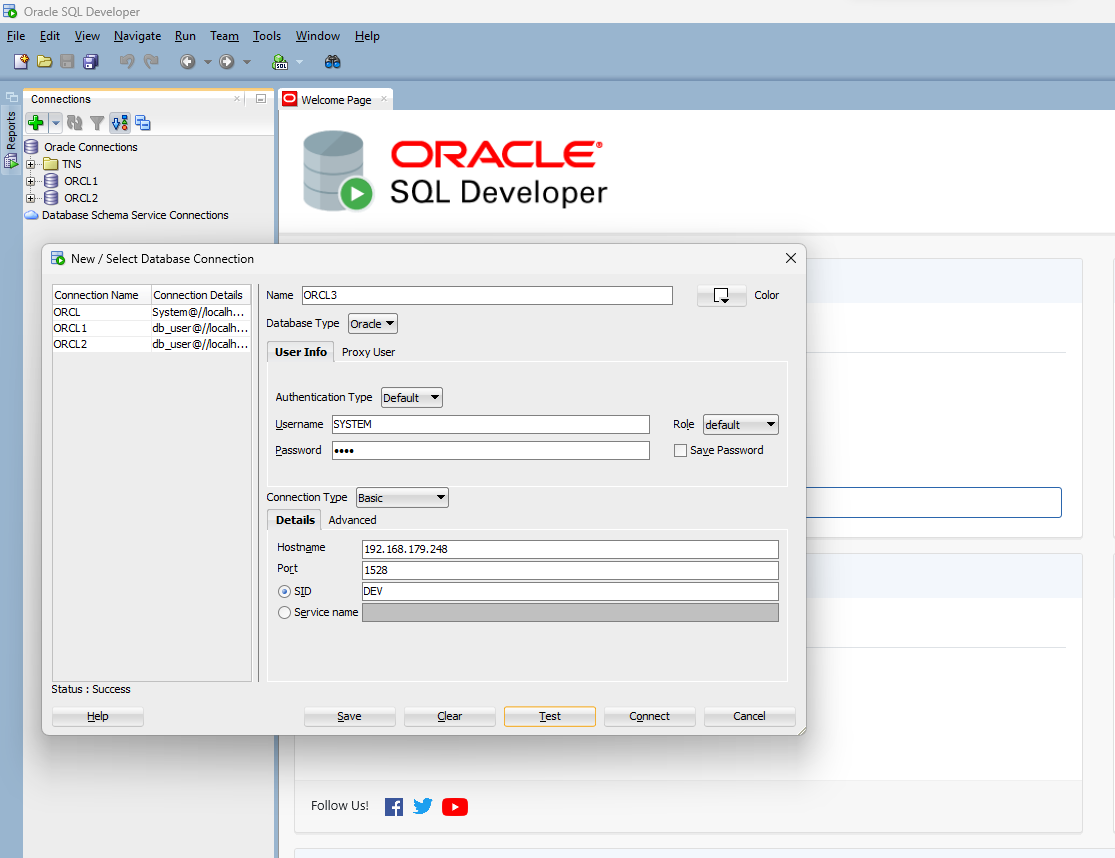


  
Datu bāzes ir veiksmīgi instalētas datorā, to var viegli pārbaudīt SQL Plus redaktorā norādot lietotājvārdu un paroli.

A screenshot of a computer

Description automatically generated

**3. Izmantojot redaktoru SQL Developer jāpieslēdzas datu bāzes sistēmai Oracle21c (jāievada savienojuma informācija). Sākumā jāizmanto iebūvētais lietotājs SYSTEM ar paroli pēc noklusējuma vai instalējot nomainīto paroli.**

Tālāk mēs dodamies uz SQL Developer redaktoru un izveidojam jaunu savienojumu ar datu bāzi.

Redzam ka viss strādā, jo Status: Succesful un sākam strādāt!

**4.  Redaktorā SQL Developer jādefinē neliela tabula (CREATE TABLE), jāievada daži tabulas raksti (INSERT) un jāveic daži datu pieprasījumi (SELECT), lai pārliecinātos, ka darba poligons strādā.**

Izveidojam nelielo tabulu PILSETAS ar comandu CREATE, ievadam 2 vērtības vienā rindā ar comandu INSERT, lai pārliecināties ka poligons darbojas.

create table PILSETAS(

PILS\_ID NUMBER PRIMARY KEY,

PILS\_NOS VARCHAR2(30)

);

A screenshot of a computer

Description automatically generated

A screenshot of a computer

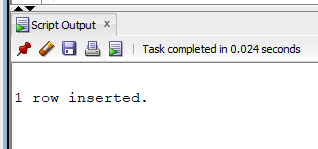
Description automatically generated

INSERT INTO PILSETAS (PILS\_ID, PILS\_NOS)

VALUES (1, 'London');

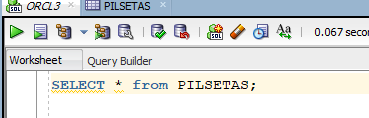
A screenshot of a computer

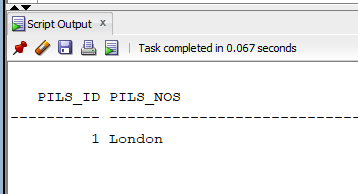
Description automatically generated



Ar komandu SELECT veicam pieprasījumu un redzam ka viss strādā, Vairāk vaicājumu būs tālāk.

SELECT \* from PILSETAS;





**5. Turpinājumā jādefinē zvaignes struktūras tabulas. Struktūru veido datu pamattabula un vismaz 3 piesaistes tabulas, kuras tiek izmantotas, lai varētu norādīt vaicājumu meklēšanas noteikumus.**

Ir izveidota tabula KLIENT

create table KLIENTI(

KLIENTA\_ID NUMBER PRIMARY KEY,

UZVARDS VARCHAR2(30),

VARDS VARCHAR2(30),

VECUMS NUMBER(3),

DZIMUMS\_F NUMBER

);

A screenshot of a computer

Description automatically generated

Ir izveidota tabula PRECES

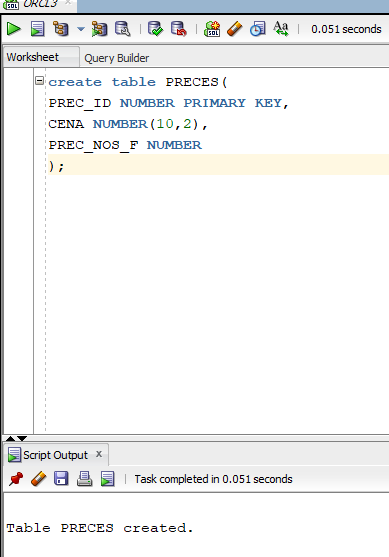
create table PRECES(

PREC\_ID NUMBER PRIMARY KEY,

CENA NUMBER(10,2),

PREC\_NOS\_F NUMBER

);



Veidojam tabulu ar nosaukumu VEIKALI.

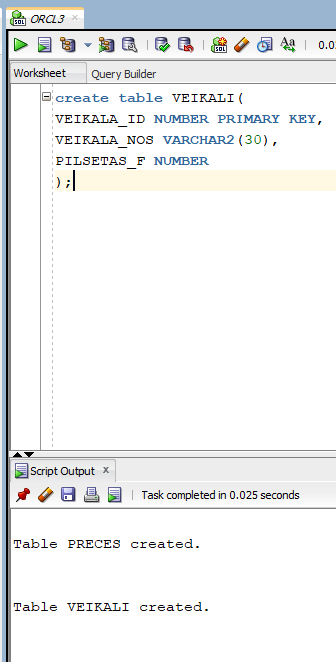
create table VEIKALI(

VEIKALA\_ID NUMBER PRIMARY KEY,

VEIKALA\_NOS VARCHAR2(30),

PILSETAS\_F NUMBER

);



Ir izveidota tabula VEIKALI

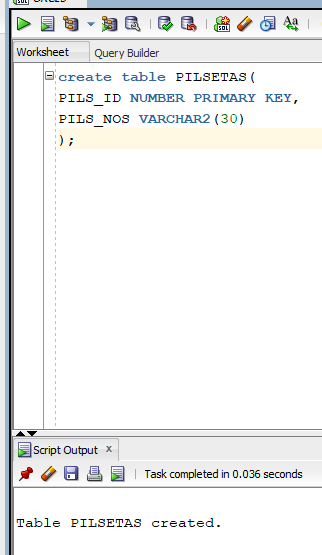
Veidojam tabulu ar nosaukumu PILSETAS.

create table PILSETAS(

PILS\_ID NUMBER PRIMARY KEY,

PILS\_NOS VARCHAR2(30)

);



Ir izveidota tabula PILSETAS

Veidojam tabulu ar nosaukumu LAIKS.

Kad sāku aizpildīt tabulu, atklāju, ka nepareizi norādījis datu tipu, tāpēc man bija jāmaina koda fragmentu un līdz ar to ekrānuzņēmumu (ekrānuzņēmums atšķiras no iepriekšējiem). Kā ir redzams, tabula ir izveidota.

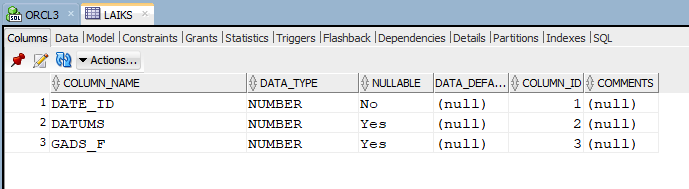
create table LAIKS(

DATE\_ID NUMBER PRIMARY KEY,

DATUMS NUMBER,

GADS\_F NUMBER

);



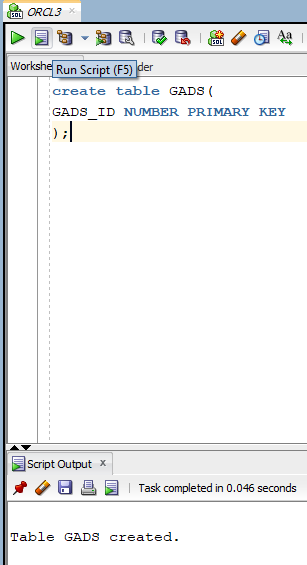
Ir izveidota tabula LAIKS.

Veidojam tabulu ar nosaukumu GADS.

create table GADS(

GADS\_ID NUMBER PRIMARY KEY

);



Ir izveidota tabula GADS.

Veidojam tabulu ar nosaukumu PASUTIJUMI.

create table PASUTIJUMI (

PAS\_ID NUMBER PRIMARY KEY,

F\_PREC\_ID NUMBER,

F\_VEIK\_ID NUMBER,

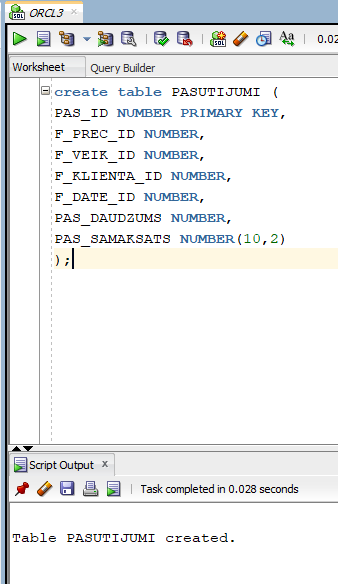
F\_KLIENTA\_ID NUMBER,

F\_DATE\_ID NUMBER,

PAS\_DAUDZUMS NUMBER,

PAS\_SAMAKSATS NUMBER(10,2)

);



Ir izveidota tabula PASUTIJUMI.

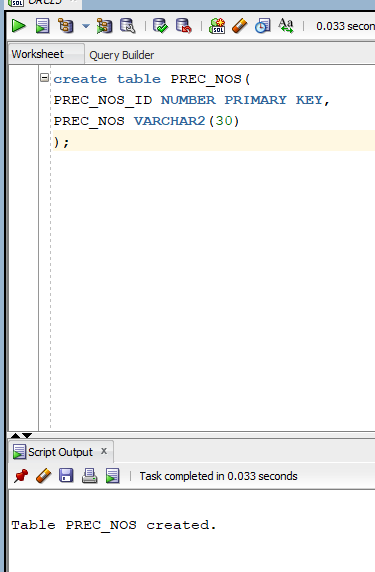
Veidojam tabulu ar nosaukumu PREC\_NOS.

create table PREC\_NOS(

PREC\_NOS\_ID NUMBER PRIMARY KEY,

PREC\_NOS VARCHAR2(30)

);



Ir izveidota tabula PREC\_NOS.

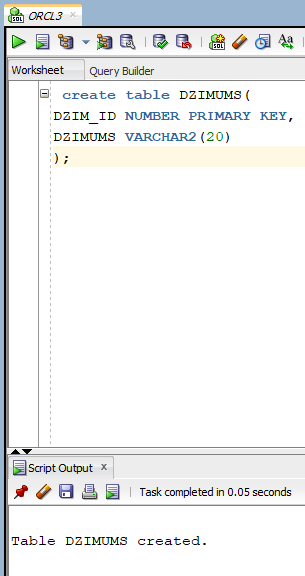
Veidojam tabulu ar nosaukumu DZIMUMS.

create table DZIMUMS(

DZIM\_ID NUMBER PRIMARY KEY,

DZIMUMS VARCHAR2(20)

);



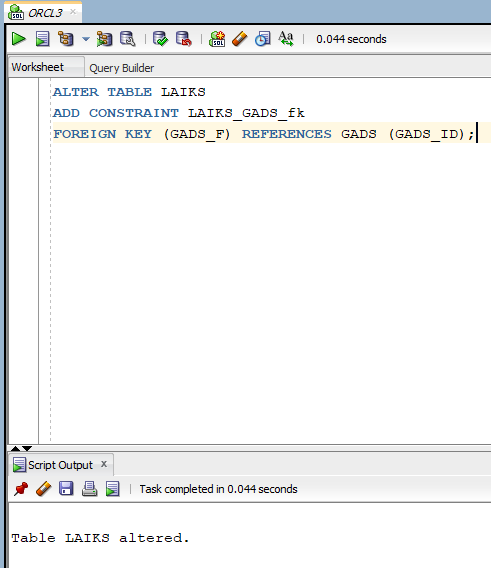
Ir izveidota tabula DZIMUMS

Šajā brīdī tabulas nav saistītas, tāpēc nākamajā solī vajag saistīt visas nepieciešamas tabulas ar ALTER TABLE.

ALTER TABLE LAIKS

ADD CONSTRAINT LAIKS\_GADS\_fk

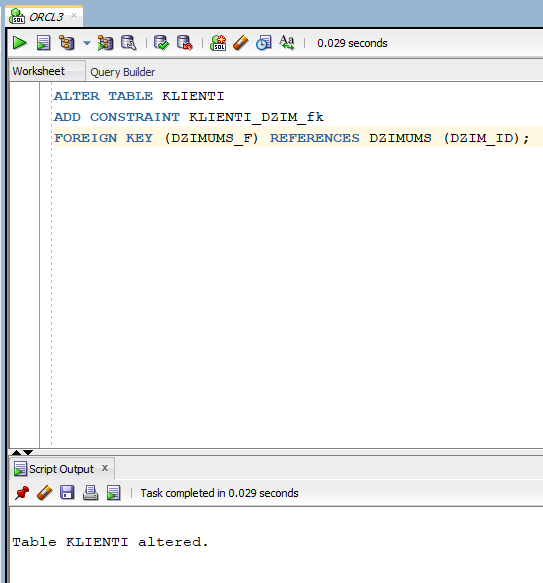
FOREIGN KEY (GADS\_F) REFERENCES GADS (GADS\_ID);



ALTER TABLE KLIENTI

ADD CONSTRAINT KLIENTI\_DZIM\_fk

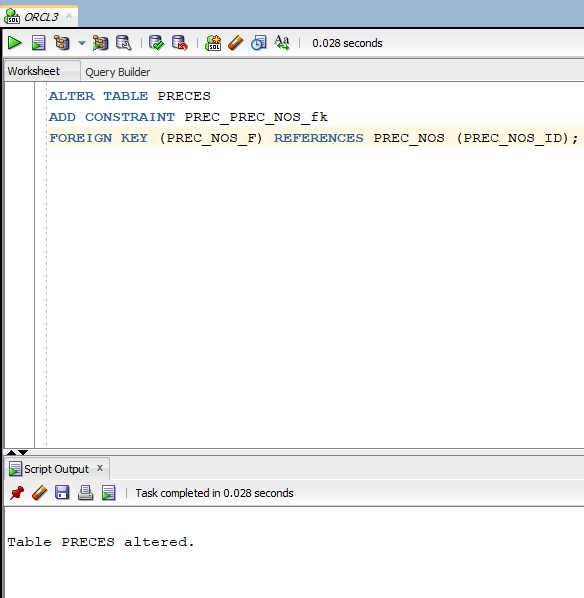
FOREIGN KEY (DZIMUMS\_F) REFERENCES DZIMUMS (DZIM\_ID);



ALTER TABLE PRECES

ADD CONSTRAINT PREC\_PREC\_NOS\_fk

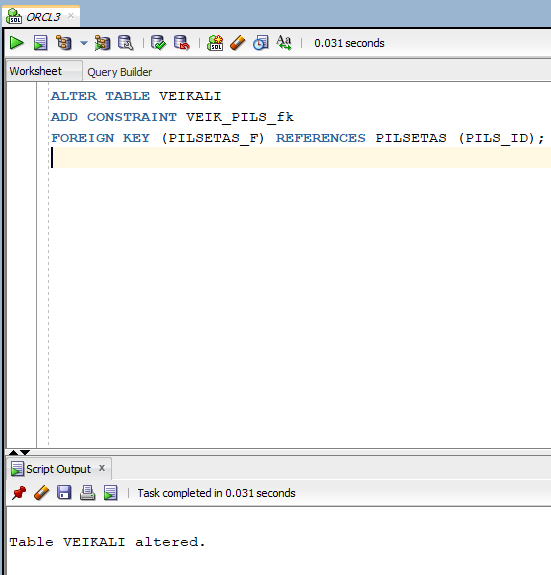
FOREIGN KEY (PREC\_NOS\_F) REFERENCES PREC\_NOS (PREC\_NOS\_ID);



ALTER TABLE VEIKALI

ADD CONSTRAINT VEIK\_PILS\_fk

FOREIGN KEY (PILSETAS\_F) REFERENCES PILSETAS (PILS\_ID);



ALTER TABLE PASUTIJUMI

ADD CONSTRAINT PAS\_PRECE\_fk

FOREIGN KEY (F\_PREC\_ID) REFERENCES PRECES (PREC\_ID)

ADD CONSTRAINT PAS\_VEIK\_fk

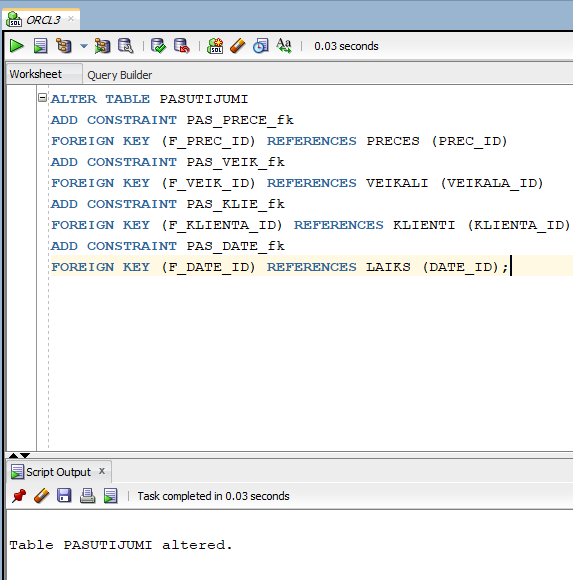
FOREIGN KEY (F\_VEIK\_ID) REFERENCES VEIKALI (VEIKALA\_ID)

ADD CONSTRAINT PAS\_KLIE\_fk

FOREIGN KEY (F\_KLIENTA\_ID) REFERENCES KLIENTI (KLIENTA\_ID)

ADD CONSTRAINT PAS\_DATE\_fk

FOREIGN KEY (F\_DATE\_ID) REFERENCES LAIKS (DATE\_ID);



Tabulas bija veiksmīgi savienotas.

**6. Pamattabulai rindu skaits ir >100 000. Tabulu aizpildīšanai ar datiem jāuzraksta speciāla PL/SQL programma, kurā datu ģenerēšanai tiek izmantoti gadījuma vērtību ģeneratori DBMS\_RANDOM. VALUE() (gadījuma skaitļi) un DBMS\_RANDOM.STRING() (gadījuma simbolu rindas). Tabulu aizpildi, jāveic ar INSERT vai UPDATE komandām.**

**Tālāk ir nepieciešams veiksmīgi tas aizpildīt. Tabulas tiek aizpildītas ar gadījuma vērtībām, kā ir teikts uzdevumā**.

/\*

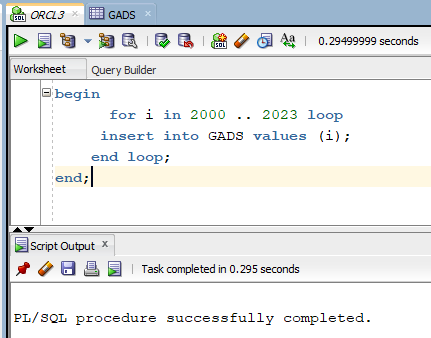
Tabulas GADS aizpildīšana ar 23 vērtībām

\*/

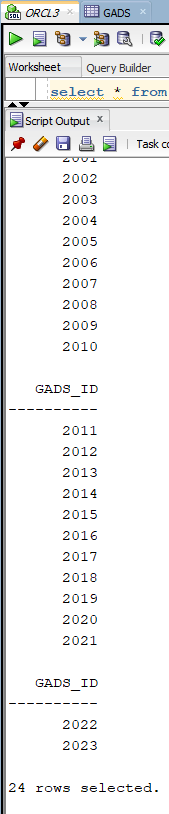
begin

for i in 2000 .. 2023 loop

insert into GADS values (i);

end loop;

end;

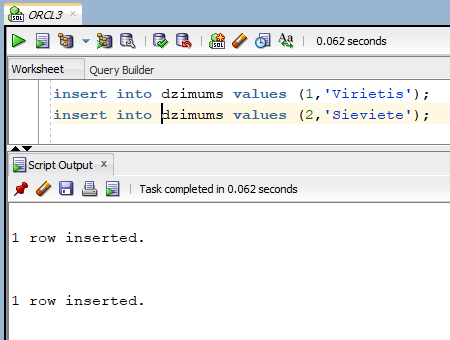


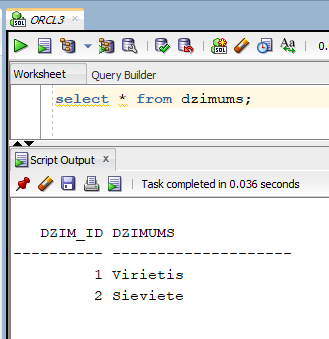
/\*

Tabulas DZIMUMS aizpildīšana ar 2 vērtībām

\*/

insert into d\_dzimums values (1,'Virietis');

insert into d\_dzimums values (2,'Sieviete');



/\*

tabulas PILSETAS aizpildīšana ar 76 vērtībām

\*/

declare i number;

type MASIVE1 is varray(76) of varchar2(15);

masivs\_pilsetas MASIVE1 := MASIVE1(

'Zilupe',

'Viļāni', 'Viļaka', 'Viesīte', 'Ainaži',

'Aizkraukle', 'Aizpute', 'Aknīste', 'Aloja', 'Alūksne', 'Ape',

'Auce', 'Baldone', 'Baloži', 'Balvi',

'Bauska', 'Brocēni', 'Cēsis', 'Cesvaine', 'Dagda', 'Daugavpils',

'Dobele', 'Durbe', 'Grobiņa', 'Gulbene',

'Ikšķile', 'Ilūkste', 'Jaunjelgava', 'Jēkabpils', 'Jelgava', 'Jūrmala',

'Kandava', 'Kārsava', 'Ķegums', 'Krāslava',

'Kuldīga', 'Lielvārde', 'Liepāja', 'Līgatne', 'Limbaži', 'Līvāni',

'Lubāna', 'Ludza', 'Madona', 'Mazsalaca',

'Ogre', 'Olaine', 'Pāvilosta', 'Piltene', 'Pļaviņas', 'Preiļi',

'Priekule', 'Rēzekne', 'Rīga',

'Rūjiena', 'Sabile', 'Salacgrīva', 'Salaspils','Saldus',

'Saulkrasti', 'Seda', 'Sigulda',

'Skrunda', 'Smiltene', 'Staicele', 'Stende', 'Strenči',

'Subate', 'Talsi', 'Tukums',

'Valdemārpils', 'Valka', 'Valmiera', 'Vangaži', 'Varakļāni',

'Ventspils');

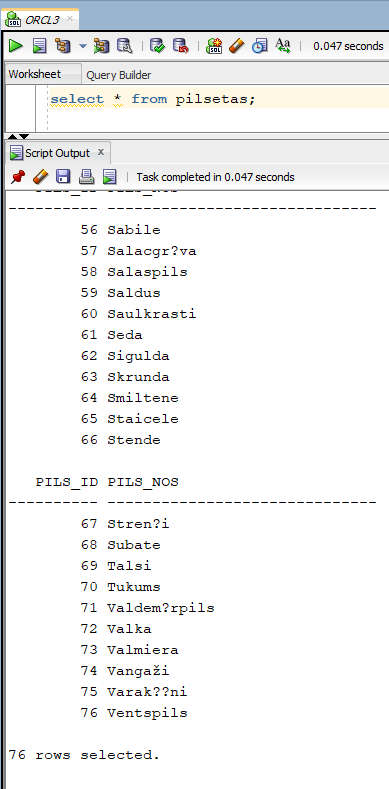
begin

for i in 1 .. 76 loop

insert into pilsetas values(i,masivs\_pilsetas(i));

end loop;

end;



/\*

tabulas PREC\_NOS aizpildīšana ar 10 000 vērtībām

\*/

declare i number;

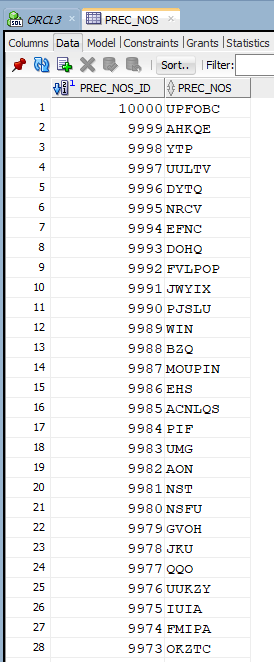
begin

for i in 1 .. 10000 loop

insert into prec\_nos values (i, DBMS\_RANDOM.string('U', TRUNC(DBMS\_RANDOM.value(3,10))));

end loop;

end;



/\*

tabulas PRECES aizpildīšana ar 100 000 vērtībām

\*/

declare i number;

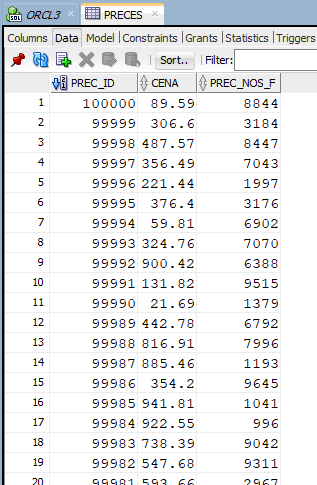
begin

for i in 1 .. 100000 loop

insert into preces values (i, DBMS\_RANDOM.value(0.50, 999.99), TRUNC(DBMS\_RANDOM.value(1, 10001)));

end loop;

end;



/\*

tabulas VEIKALI aizpildīšana ar 1 000 vērtībām

\*/

declare i number;

begin

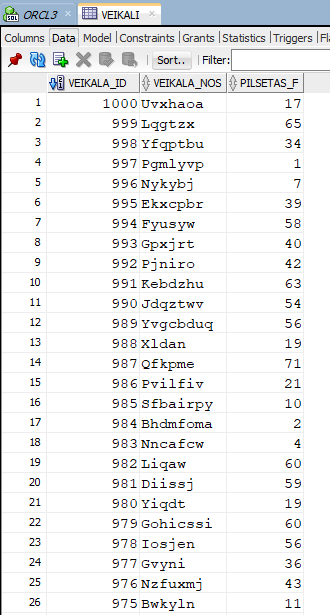
for i in 1 .. 1000 loop

insert into veikali values (i, DBMS\_RANDOM.string('U', 1) ||

DBMS\_RANDOM.string('L', TRUNC(DBMS\_RANDOM.value(4,8))), TRUNC(DBMS\_RANDOM.value(1, 77)));

end loop;

end;



/\*

tabulas KLIENTI aizpildīšana ar 100 000 vērtībām

\*/

declare

i number;

--Vīriešu vārdu nosaukumi

type MASIVE4 is varray(56) of varchar2(20);

masivs\_vards\_viriesu MASIVE4 := MASIVE4(

'Antons','Natans', 'Andrejs','Kirils','Roberts', 'Aivars',

'Maks', 'Dans', 'Romans', 'Arturs',

'Stepans','Valerijs', 'Ričiks', 'Miša', 'Valentins', 'Kristaps',

'Lacis', 'Daniils', 'Sergejs', 'Eriks',

'Filips', 'Zigmunds', 'Aleksejs', 'Kostja', 'Alex', 'Viliams',

'Bruno', 'Deniss', 'Dmitrijs', 'Artjoms',

'Gints', 'Renars', 'Raivo', 'Ruslans', 'Viliams', 'Ilja',

'Arsenijs', 'Zahars', 'Jans', 'Janis',

'Ignats', 'Vlads', 'Georgijs', 'Peteris', 'Amirs', 'Emils',

'Rustams', 'Mihails', 'Egors', 'Matvejs',

'Vitalijs', 'Bogdans', 'Juris', 'Vadims', 'Savelijs', 'Boris');

--Sieviešu vārdu nosaukumi

type MASIVE5 is varray(60) of varchar2(20);

masivs\_vards\_sieviesu MASIVE5 := MASIVE5(

'Marija','Māra', 'Natali', 'Vlada','Gunta','Sintija', 'Dasha',

'Aida','Natasha','Vika','Anna','Dace',

'Agita', 'Sofija','Vaida', 'Luiza',

'Luize','Kira','Mona', 'Dina', 'Alina','Maiga',

'Laima', 'Linda','Krispa', 'Melanija',

'Mira', 'Iga', 'Lora', 'Ieva',

'Alise', 'Koko', 'Manita', 'Darine', 'Erika',

'Laura','Nadežda', 'Vika', 'Sasha', 'Alla','Mara', 'Mina', 'Roza',

'Kristina', 'Aljona','Olga', 'Andzela', 'Vera', 'Madona', 'Liene',

'Kristi','Avita', 'Ieva', 'Liesma', 'Alexandra', 'Karlīna');

--Uzvārdi

type MASIVE6 is varray(10000) of varchar2(20);

masivs\_uzvards MASIVE6 := MASIVE6();

--Dzimumi

type MASIVE7 is varray(2) of number(2);

masivs\_dzimums MASIVE7 := MASIVE7(1,2);

begin

--Uzvārdu saģenerēšana masīvā

for i in 1 .. 10000 loop

masivs\_uzvards.extend;

masivs\_uzvards(i) := DBMS\_RANDOM.string('U', 1) ||

DBMS\_RANDOM.string('L', TRUNC(DBMS\_RANDOM.value(3,15)));

end loop;

for i in 1 .. 100000 loop

if masivs\_dzimums(TRUNC(DBMS\_RANDOM.value(1, 3))) = 2

then

insert into KLIENTI values(i,

masivs\_uzvards(TRUNC(DBMS\_RANDOM.value(1, 10000))),

masivs\_vards\_sieviesu(TRUNC(DBMS\_RANDOM.value(1, 56))),

TRUNC(DBMS\_RANDOM.value(18, 75)),

2);

else

insert into KLIENTI values(i,

masivs\_uzvards(TRUNC(DBMS\_RANDOM.value(1, 10000))),

masivs\_vards\_viriesu(TRUNC(DBMS\_RANDOM.value(1, 56))),

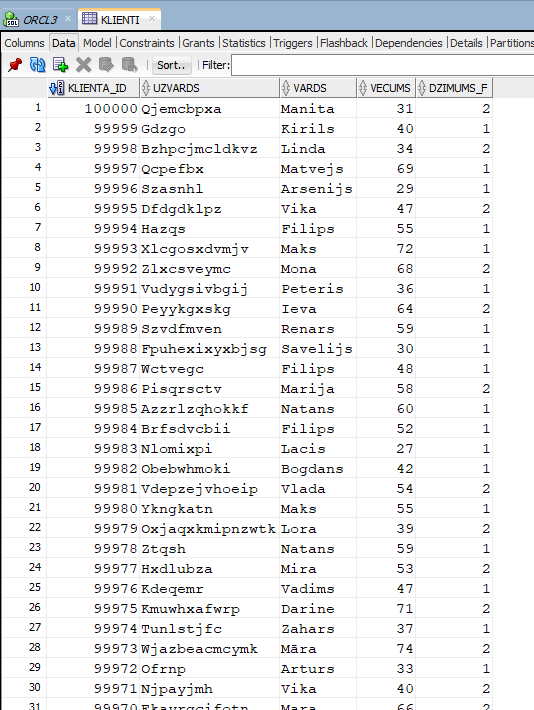
TRUNC(DBMS\_RANDOM.value(18, 75)),

1);

end if;

end loop;

end;



/\*

tabulas LAIKS aizpildīšana ar 100 000 vērtībām

\*/

declare i number;

begin

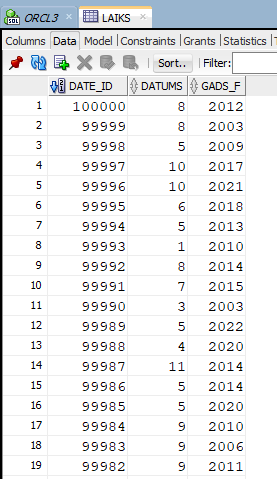
for i in 1 .. 100000 loop

insert into LAIKS values

(i, TRUNC(DBMS\_RANDOM.value(1, 12)), TRUNC(DBMS\_RANDOM.value(2000, 2023)));

end loop;

end;



/\*

tabulas PASUTIJUMI aizpildīšana ar 200 000 vērtībām

\*/

declare i number;

n number;

k number;

begin

for i in 1 .. 200000 loop

n := TRUNC(DBMS\_RANDOM.value(1, 100001));

k := TRUNC(DBMS\_RANDOM.value(1, 31));

insert into PASUTIJUMI values (

i,

n,

TRUNC(DBMS\_RANDOM.value(1, 1001)),

TRUNC(DBMS\_RANDOM.value(1, 100001)),

TRUNC(DBMS\_RANDOM.value(1, 100001)),

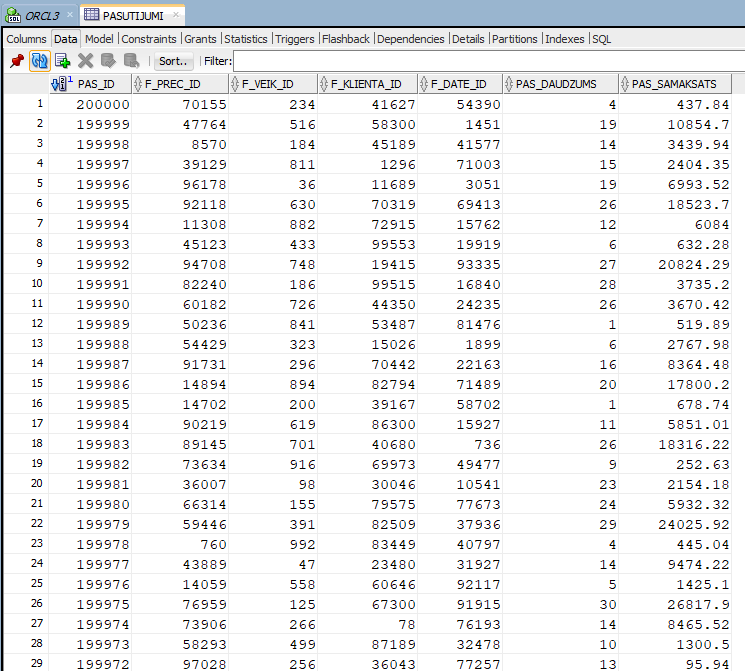
k,

k \* (SELECT preces.cena from preces WHERE preces.prec\_id = n)

);

end loop;

end;



Visas tabulas bija veiksmīgi aizpildītas.

**7. Tabulu pārbaudei jāuzraksta un jāizpilda 3 SQL vaicājumi pamattabulas datu agregātvērtību SUM(), COUNT(), AVG() iegūšanai. Meklēšanas noteikumi tiek norādīti, izmantojot pakārtotās tabulas.**

**Šis vaicājums aprēķina kopējo summu, ko klients iztērējis.**

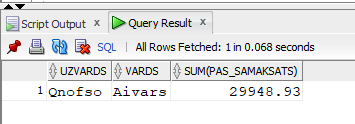
SELECT UZVARDS, VARDS, SUM(PAS\_SAMAKSATS)

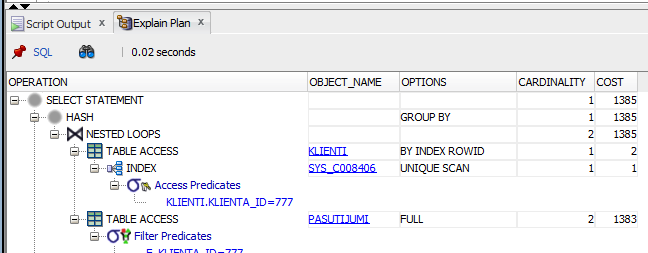
FROM KLIENTI

JOIN PASUTIJUMI ON KLIENTA\_ID = F\_KLIENTA\_ID

WHERE KLIENTA\_ID = 777

GROUP BY UZVARDS, VARDS;





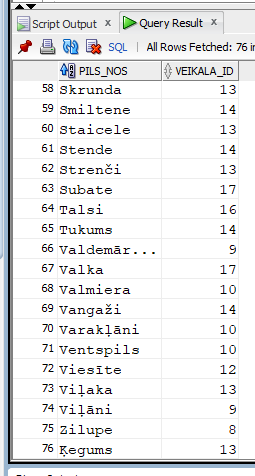
**Šis vaicājums uzskaita veikalu skaitu noteiktā pilsētā**

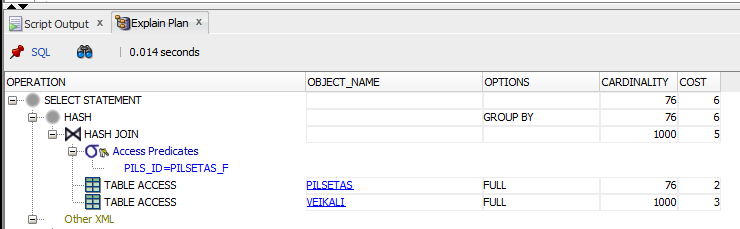
SELECT PILS\_NOS , COUNT(VEIKALA\_ID) as VEIKALA\_ID

FROM PILSETAS

JOIN VEIKALI ON PILS\_ID = PILSETAS\_F

GROUP BY PILS\_NOS;



****

**Šis vaicājums aprēķina vidējo veikalu skaitu, Rīgā**

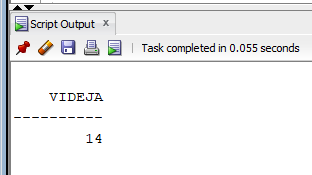
SELECT AVG(COUNT(V.VEIKALA\_ID)) as Videja

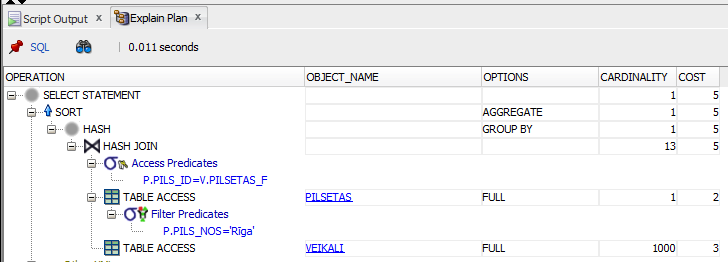
FROM PILSETAS P

JOIN VEIKALI V ON P.PILS\_ID = V.PILSETAS\_F

WHERE PILS\_NOS = 'Rīga'

GROUP BY PILS\_NOS;





**SECINĀJUMI**

Izpildot darbu līdz beigām, var pateikt, ka bija definēta, zvaigznes struktūras tabulas kur arī ir faktu tabula un 4 piesaistes tabulas. Tā ka darbs nav nodots paredzētajā laika, es pārskatīju nākamos uzdevumus un uzreiz uztaisīju struktūru ar kuru strādāšu nākamajos praktiskos darbos. Runājot par DB instalēšanu, ilgu laiku man bija problēmas to atjaunot līdz pēdējai versijai, tāpēc ka es aizmirsu SYS paroli. Galu galā es uzinstalēju jauno, un tagad viss strādā.

Daži vārdi par vaicājumiem, mēs vel daudz runāsim par viņiem, bet uzreiz var apskatīt pirmos trīs un ir redzams, ka viens no viņiem pārskata FULL tabulu ar faktiem (PASUTIJUMI) un tām vaicājumam COST vērtība ir lielāka nekā citam. Bet abi citi ir vienkārši un pārskata nelielas datu tabulas.