

RĪGAS TEHNISKĀ UNIVERSITĀTE

Datorzinātnes un informācijas tehnoloģijas fakultāte

4.praktiskais darbs

mācību priekšmetā

“Datu bāzu pamati un Relāciju datu bāzes (DB1)”

**SQL valodas paplašinājumi**

Izstrādāja: Viktors Stepičevs

II.kurss, 7.grupa, 191RDB100

Pārbaudīja: lektors A.Auziņš

**2020./21/māc.gads**

**Darba saturs**

[**Darba uzdevumi** 3](#_Toc71488707)

[**Anotācija** 4](#_Toc71488708)

[**Tabulu struktūru definēšana** 5](#_Toc71488709)

[**Vaicājumu definēšana un pārbaudīšana** 7](#_Toc71488710)

[**Hierarhiskie vaicājumi** 7](#_Toc71488711)

[**Faktoru analīzes vaicājumi vienai tabulai** 9](#_Toc71488712)

[**Vaicājumi tabulu zvaigznes shēmai** 15](#_Toc71488713)

[**Virtuālas daudzdimensiju tabulas vaicājumi ar SQL Model klauzulas izmantošanu** 17](#_Toc71488714)

[**Vaicājumi ar analīzes funkciju izmantošanu** 20](#_Toc71488715)

[**Vaicājumi ar loga (windowing) funkciju izmantošanu** 22](#_Toc71488716)

[**Vaicājumi ar WITH priekšraksta izmantošanu** 23](#_Toc71488717)

[**Rekursīvie vaicājumi ar WITH priekšraksta izmantošanu** 24](#_Toc71488718)

[**Secinājumi** 26](#_Toc71488719)

[**Literatūra** 27](#_Toc71488720)

[**Pielikums** 28](#_Toc71488721)

[**Darbā izmantotas datu bāzes SQL kods** 28](#_Toc71488722)

# **Darba uzdevumi**

**Katrai vaicājumu grupai izmantot atbilstošu tabulu struktūru, veikt datu ievadi, definēt vaicājumus, iegūt rezultātus, veikt rezultātu analīzi, lai parādītu, ka dotais vaicājuma tips izprasts.**

1. **Hierarhiskie vaicājumi.**
   * Izveidot vaicājumus izmantojot START WITH, CONNECT BY un LEVEL atslēgvārdus.
2. **Faktoru analīzes vaicājumi vienai tabulai.**
   * Izveidot vienu tabulu ar vismaz 3 mainīgiem faktoriem un rezultējošo faktu.
   * Izveidot vaicājumus izmantojot atslēgvārdus GROUP BY CUBE, GROUPING(), GROUPING SETS.
3. **Vaicājumi tabulu zvaigznes shēmai.**
   * Izveidot vismaz 3 dimensiju tabulas un faktu tabulu.
   * Izveidot vaicājumus izmantojot atslēgvārdus GROUP BY CUBE, GROUPING(), GROUPING SETS.
4. **Virtuālas daudzdimensiju tabulas vaicājumi ar SQL Model klauzulas izmantošanu.**
   * Izveidot vienu tabulu ar vismaz 3 mainīgiem faktoriem un rezultējošo faktu.
   * Izveidot vaicājumus izmantojot atslēgvārdus MODEL, PARTITION BY, DIMENSION BY, MEASURES, RULES.
5. **Vaicājumi ar analīzes funkciju izmantošanu.**
   * Izveidot vaicājumus izmantojot atslēgvārdus OVER, PARTITION BY, ORDER BY, RANK, DENSE\_RANK, ROW\_NUMBER, FIRST, LAST, NTILE, WIDTH\_BUCKET.
6. **Vaicājumi ar loga (windowing) funkciju izmantošanu.**
   * Izveidot vaicājumus izmantojot atslēgvārdus ROWS, RANGE, BETWEEN, UNBOUNDED, PRECEDING, FOLLOWING, CURRENT ROW.
7. **Vaicājumi ar WITH priekšraksta izmantošanu.**
   * Izveidot vaicājumus izmantojot atslēgvārdus WITH, UNION ALL.
8. **Rekursīvie vaicājumi ar WITH priekšraksta izmantošanu.**
   * Izveidot vaicājumus izmantojot atslēgvārdus WITH, UNION ALL, SEARCH DEPTH.

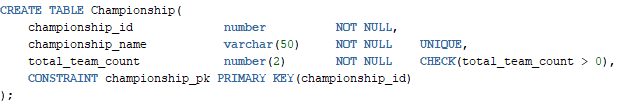
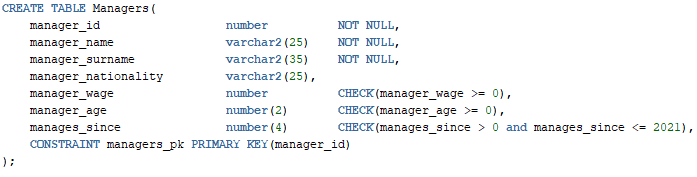
# **Anotācija**

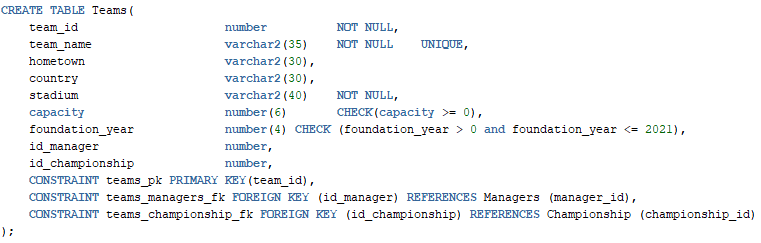
Darba uzdevumu izpildes laikā tiks definēti un pārbaudīti vairākas tabulu struktūras un vārākie dažādu tipa vaicājumi. Definējot vaicājumus pēc noteikta tipa tiks aplūkotas vairākas funkcijas un konstrukcijas, kuras dod vairākas iespējas vaicājumu realizācijai un izprašanai. Darbs tika izstrādāts ar Oracle SQL Developer rīka palīdzību.

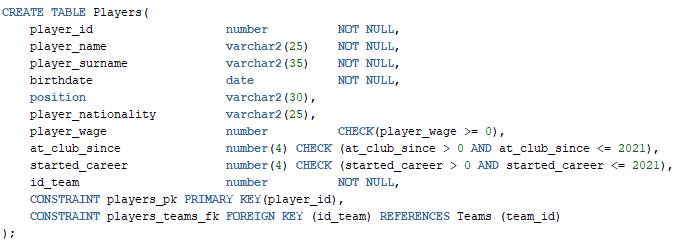
# **Tabulu struktūru definēšana**

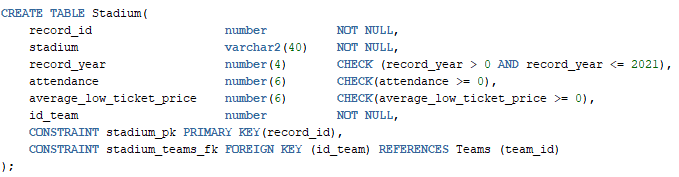
Pirms uzdevumu izpildes uzsākšanas vispirms ir jādefinē tabulu struktūras. Par pamatu šī darba izveidei es izmantoju tabulas no iepriekšējiem darbiem (Managers, Teams, Players), bet ar tiem, lai izpildītu visus darba uzdevumus nepietiek. Tādēļ, esošās tabulas tika papildinātas ar dažiem atribūtiem, kā arī tika izveidotas divas jaunas tabulas “Championship” un “Stadium”. Tabulai “Managers” tika pievienots trenera atalgojums par nedēļu “manager\_wage” atribūts, kura datu tips ir “number” ar “nenegavitātes” pārbaudi. Savukārt, ar tādu pašu principu tabulai “Players” tika pievienots spēlētāja atalgojuma par nedēļu atribūts “player\_wage”.

Jaunajā tabulā “Championship” glabājas informācija par turnīriem, kuros piedalās komandas. Par primāro atslēgu izveidoju atribūtu “championship\_id”, kurš ir referencēts ar atribūtu “id\_championship” tabulā “Teams”. Atribūts “championship\_name” ir turnīra nosaukums un atribūts “total\_team\_count” ir kopējais komandu skaits, kurš piedalās turnīrā. Tabulā “Stadium” glabājās apkopojoša statistika par stadioniem. Ieraksta identifikators(tabulas PK) “record\_id”, stadiona nosaukums “stadium”, ieraksta gads “record\_year”, stadiona apmeklējums “attendance”, “lētāko biļešu vidēja” cena(vidēja cena no lētākajiem biļetēm atkarībā no vairākiem parametriem) “average\_low\_ticket\_price” un ārējā atslēga “id\_team”.









Kopumā, visi ievades dati, kuri ir ievadīti tabulās ir akurāti vai tuvi reālajiem. Nobīde no realitātes var rasties avotu dažādības dēļ. Ievades dati, kā arī pārējais darba kods varētu būt atrodams atskaites un iesnieguma pielikumos.

**Vaicājumu definēšana un pārbaudīšana**

## **Hierarhiskie vaicājumi**

1. **Vaicājums**



Pirmais vaicājums izvada informāciju par spēlētāju un tā komandu ar CONNECT BY palīdzību. NOCYCLE parametrs dod instrukciju datu bāzei atgriezt rindas pat tad, kad veidojas cilpa. Kā redzams rezultātā, pēc 17. spēlētāja spēlētāji sāka atkārtoties no jauna. Pēc tam, kad visi spēlētāji tika izvadīti, tikai tad saraksts turpinājās ar 18. spēlētāju, kopumā izvadot katru spēlētāju divas reizes. (Neievietoju visus vaicājuma rezultātus, jo būtiski tas izvada katru spēlētāju divas reizes un kopumā principu aprakstīju augstāk)

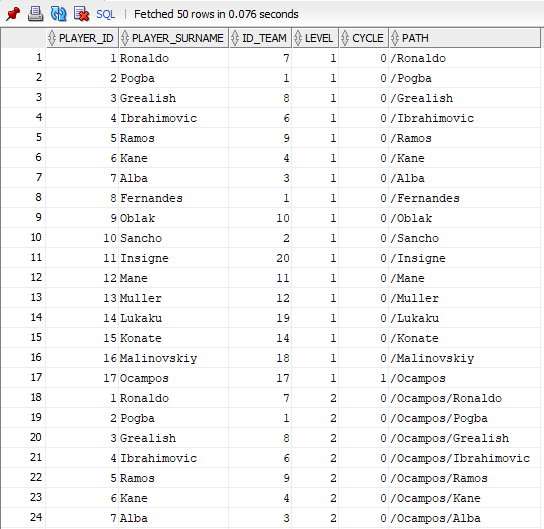


…

1. **Vaicājums**

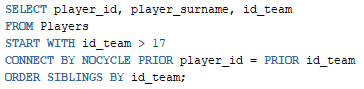
****

Papildinot vaicājumu ar dažam funkcijām ir iespējams detalizēt saprast rezultātus. Ar funkcijas LEVEL palīdzību vaicājums izvada katra spēlētāja “hierarhijas” līmeni. CONNECT\_BY\_ISCYCLE rezultātā atgriež ‘1’ tajā ierakstā, kur veidojas cilpa, bet SYS\_CONNECT\_BY\_PATH() atgriež hierarhijas ceļu. Skatoties uz pirmā vaicājuma rezultātus tagad ir saprotams un redzams, ka patiešām cilpa veidojas 17. spēlētāja rindā.

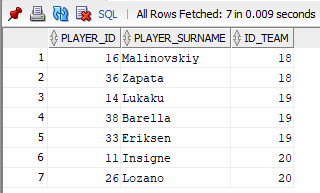
****

…

1. **Vaicājums**

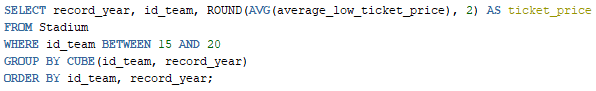


Ar šo vaicājumu atšķirība no citiem izvadīju informāciju tikai par dažiem spēlētājiem(ar komandas ID > 17), lai rezultāti būtu novērojamāki. Ar ORDER SIBLINGS BY konstrukciju tika izvadīti hierarhijas līmeņa secībā.

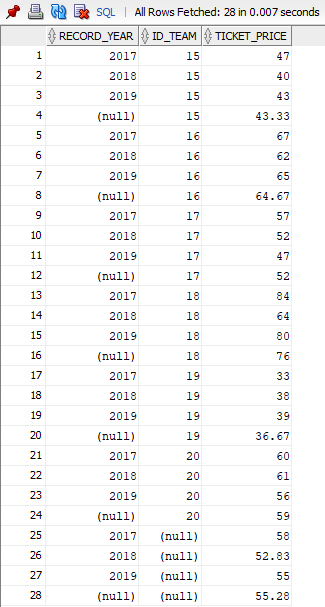
****

## **Faktoru analīzes vaicājumi vienai tabulai**

1. **Vaicājums**

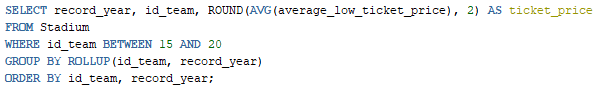


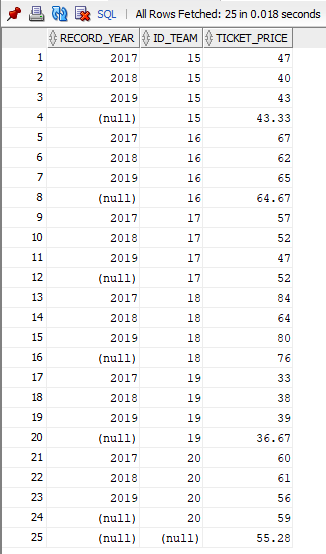
Ar pirmā vaicājuma palīdzību tika izvadīta informācija par biļešu cenu komandām ar ID no 15 līdz 20. GROUP BY() klauzulas paplašinājums CUBE izvada ģenerētas aprēķinu starpvērtības. Šī vaicājuma kontekstā tika aprēķināta vidēja biļešu cena triju gadu diapazonā katrai komandai, kā arī tika aprēķināta vidēja biļešu cena visām komandām atkarībā no gada un vidēja vērtība visiem atlasītiem ievades datiem. Vidējās vērtības tika noapaļotas līdz 2 zīmēm aiz komata, ņemot vērā, ka tā ir biļešu cena.



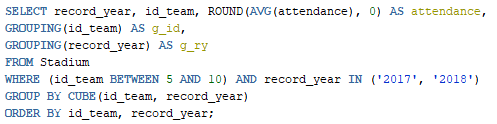
Kopumā, gandrīz to pašu informāciju ir iespējams iegūt izmantojot paplašinājumu ROLLUP, nevis CUBE, bet tomēr tas neģenerē vidējas starpvērtības visu dimensiju ierakstiem. Kā būs redzams piemērā, ROLLUP neģenerēja kopēju vidēju cenu visām komandām pa gadiem.

* 1. **Vaicājums**

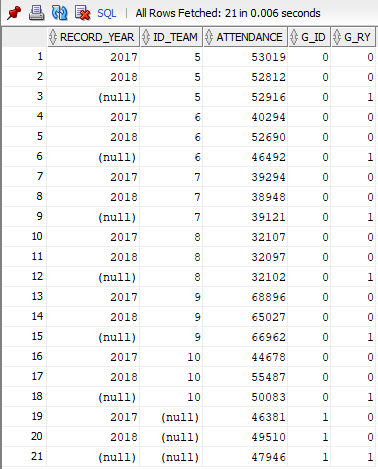




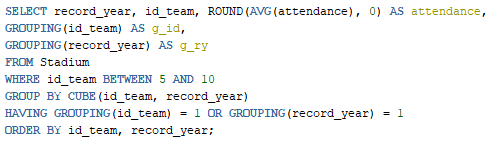
1. **Vaicājums**

****

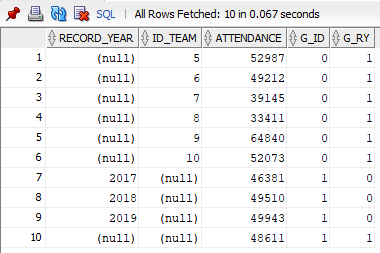
Otrajā vaicājumā tika atrasta stadiona apmeklējuma statistika 2017. un 2018. gados komandām ar ID no 5 līdz 10. Kā arī pirmajā vaicājumā, GROUP BY CUBE() izvada vidējas starpvērtības katrai komandai un kopsavilkumu beigās. Izmantojot GROUPING() funkciju ir iespējams novērot pēc kura parametra tika ģenerēta starpvērtība. Vidējas vērtības noapaļoju līdz veselam skaitlim, ņemot vērā kā ar apmeklējumu sastāda cilvēki.

****

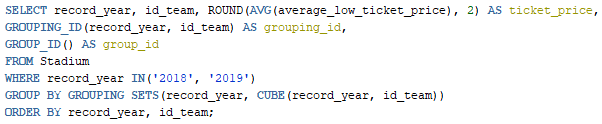
1. **Vaicājums**

****

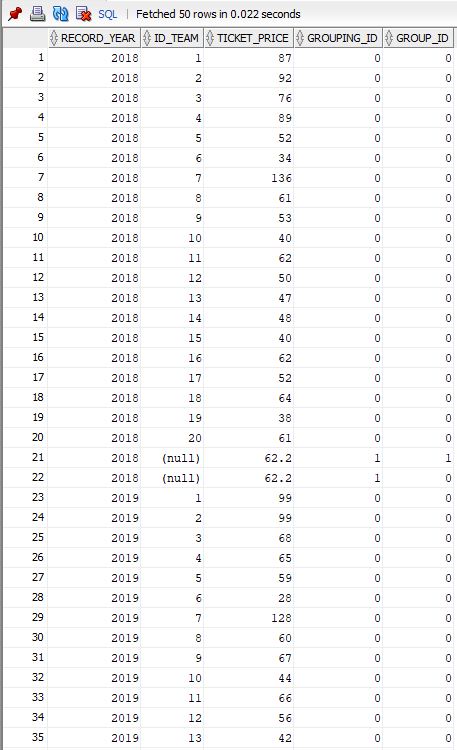
Pievienojot papildus GROUPING() nosacījumus HAVING klauzulai ir iespējams izvadīt tikai tās vidējas vērtības, kuras tika ģenerētas ar GROUP BY CUBE() konstrukcijas palīdzību. Tas tiek realizēts definējot vismaz vienam no GROUPING() nosacījumiem vērtību ‘1’, kas nozīmē, ka rezultātā tiks izvadītas tikai ar GROUP BY CUBE() ģenerētas rindas. Ar šī vaicājuma palīdzību tika atrastas stadiona apmeklējuma vidēja vērtība triju gadu laikā komandām ar ID no 5 līdz 10.

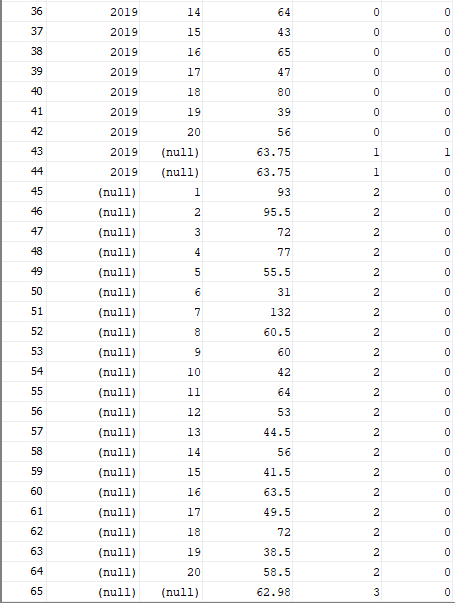
****

1. **Vaicājums**

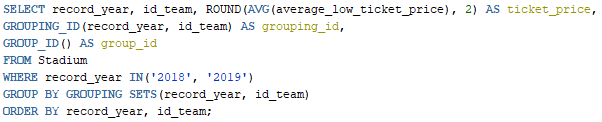
****

Šajā vaicājumā tika izmantots GROUP BY paplašinājums GROUPING SETS() kombinācijā ar paplašinājumu CUBE(), grupējot datus pēc ieraksta gadu un arī izvadot CUBE ģenerēto starpvērtības. Kā arī ir redzams, šis vaicājums izvada divas vienādas rindas ar ģenerētam vidējam vērtībām. Izmantojot GROUP\_ID() funkciju kombinācijā ar GROUPING\_ID() funkciju ļauj analizēt pēc kurā principā izvadītā rinda tika sagrupēta (to dubulto vidējo vērtību izvadi ir iespējam novērst, pievienojot nosacījumu, ka GROUP\_ID ir vienāds ar ‘0’). Izvadot visu “ievades datu” informāciju un vidējo vērtību katram gadam, pēc tām tika izvadīta vidēja biļešu cena katrai komandai 2018. un 2019. gada periodā.

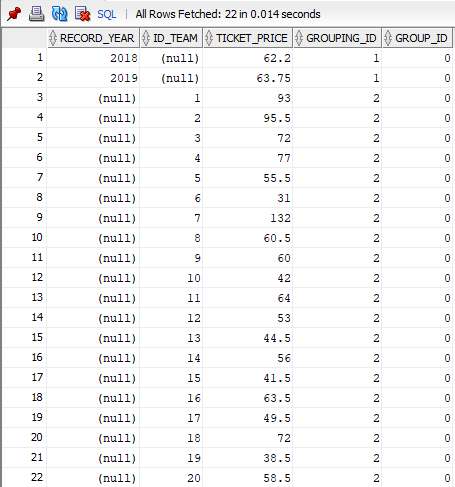
****

****

* 1. **Vaicājums**

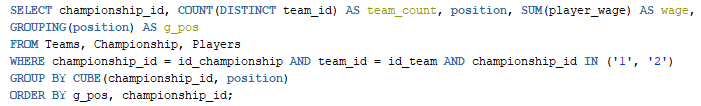
****

Grupējot datus izmantojot GROUPING SETS() paplašinājumu bez CUBE pielietojuma ir redzams, kā dati tiek secīgi klasificēti pēc definētājiem parametriem. Kopumā, vienīgā atšķirība ir tas, ka šīs vaicājums neatgriež visu “ievades” statistiku, bet rezultē tikai ar aprēķinātām vidējām vērtībām katram no atribūtu definētājām vērtībām. Arī ir redzams, ka funkcija GROUP\_ID() visur atgriež ‘0’ vērtību.

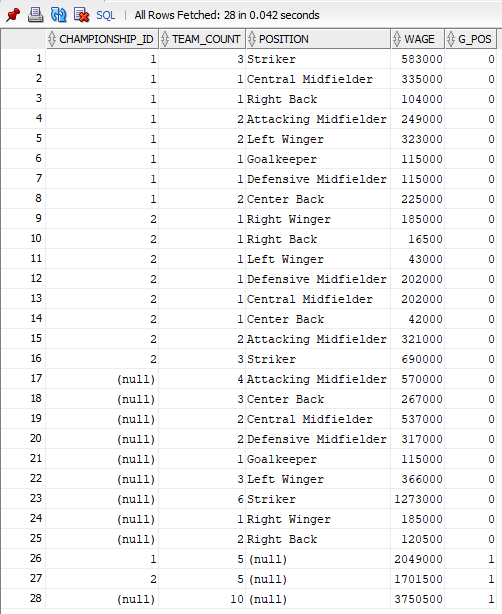
****

## **Vaicājumi tabulu zvaigznes shēmai**

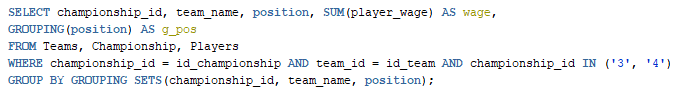
1. **Vaicājums**

****

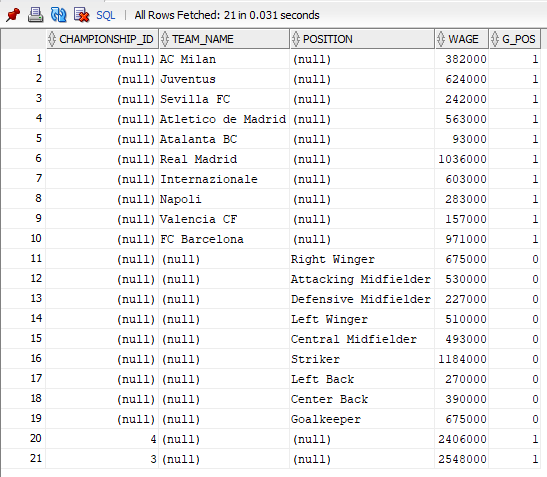
Šī vaicājuma princips kopumā ir līdzīgs, kā iepriekšējā vaicājumu grupā, bet šajā gadījumā tas ir pielietots zvaigznes tabulu shēmai (Tabulas Teams, Championship un Players). Ar šo vaicājumu tika sagrupēti visi spēlētāji atkarībā no nacionāla turnīra un pozīcijas, atradot cik komandās no līgām ar ID 1 UN 2 spēlē noteikta profila spēlētāji, katram ierakstam sasummējot atalgojumu. Komandas tika saskaitītas ar COUNT(DISTINCT) funkcijas palīdzību, jo šajā kontekstā neizmantojot DISTINCT funkcija atgriezis visu ierakstu skaitu, kurš būs līdzīgs ar spēlētāju skaitu, bet mums ir nepieciešamas tikai “īpašas” vērtības.



1. **Vaicājums**

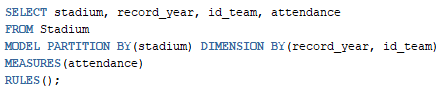
****

Pēc līdzīga principa kā iepriekš, šajā vaicājumā tika atgriezta spēlētāju atalgojuma kopējas vērtības komandām no turnīra ar ID 3 un 4 atkarībā no pati komandas, turnīrā un spēlētāju pozīcijas.

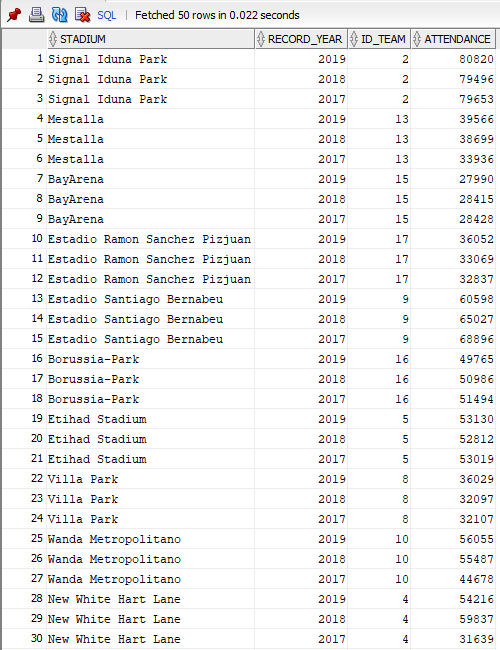
****

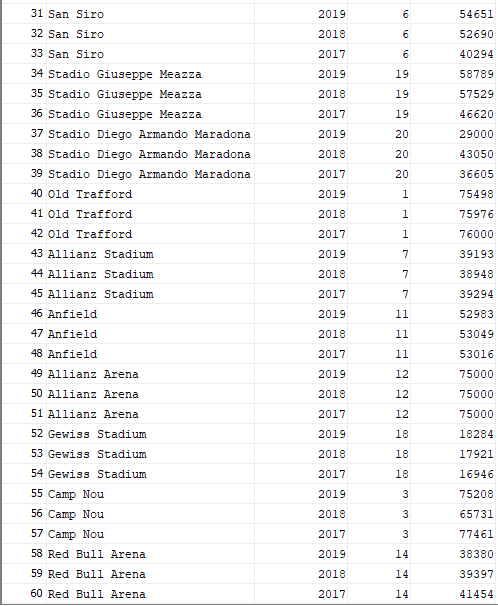
## **Virtuālas daudzdimensiju tabulas vaicājumi ar SQL Model klauzulas izmantošanu**

1. **Vaicājums**

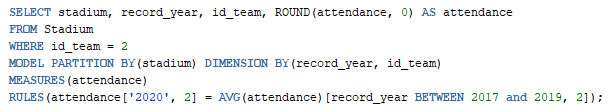


Šī vaicājuma jēga ir izvadīt statistisko informāciju par stadionu aizņemtību. Kā redzams, ar MODEL klauzulas palīdzību izmantojot funkciju PARTITION BY() dati tika izvadīti secīgi, izvadot visu pieejamo informāciju par stadionu kopā. Papildus nosacījumi pie RULES() klauzulas netika definēti.

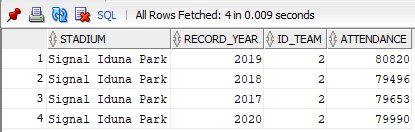




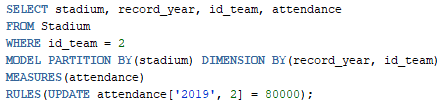
1. **Vaicājums**

****

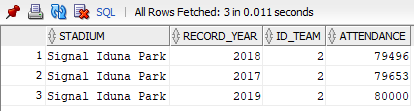
No iepriekšējā vaicājuma šīs atšķiras ar to, ka tika pievienoti papildus nosacījumi pie RULES() klauzulas. Lai samazinātu izvades datu daudzumu pamēģināju pārbaudīt RULES() klauzulas darbību uz vienas komandas, pievienojot jauno ceturto ierakstu ar gada vērtību 2020 un apmeklējuma vērtību līdzīgo ar vidējo apmeklējumu skaitu no iepriekšējo triju gadu statistikas.

****

1. **Vaicājums**

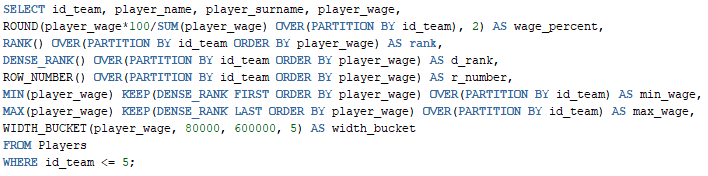
****

Līdzīgi iepriekšējām vaicājumam tika iestādīts nosacījums RULES() klauzulā, tikai šoreiz tika atjaunots jau esošais 2019. gada ieraksts, kuram tika atjaunota apmeklējuma vērtība uz 80000 ar funkciju UPDATE.

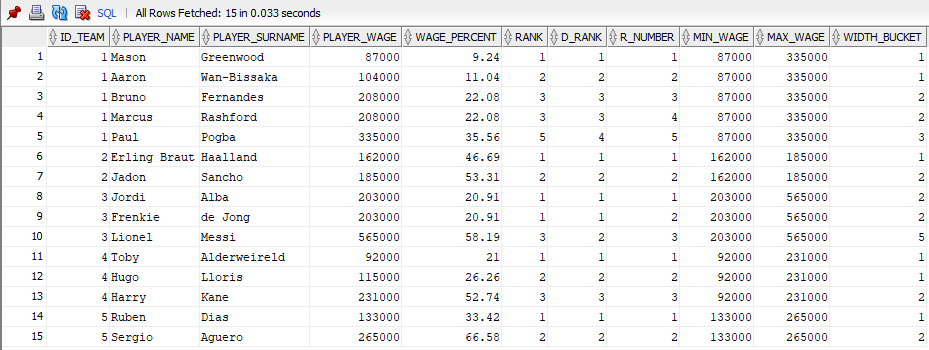
****

## **Vaicājumi ar analīzes funkciju izmantošanu**

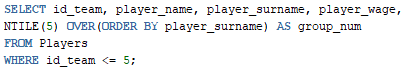
1. **Vaicājums**



Sekojoša vaicājuma ietvaros iekļāvu vairākas funkcijas, lai varētu novērojami aplūkot to funkciju darbības pamatprincipus un atšķirības. Vispirms, aprēķināju atalgojuma procentuālo attiecību katram spēlētājam. Ar OVER(PARTITION BY) konstrukciju definēju, ka šo attiecību ir jāaprēķina vienas komandas ietvaros. Funkcijām RANK() un DENSE\_RANK() kopumā ir līdzīgas īpašības. Tās ir izmantotas datu sakārtošanai. Vienīga atšķirība tām funkcijām ir tas, ka DENSE\_RANK() sastopot līdzīgas vērtības klasificē tās zem viena ranga un pēc tam, kad tika sastapta nākamā vērtība, secīgi parej uz nākamo skaitlisko vērtību, bet RANK() to secību palaid un izvēlās nākamo. Funkcija ROW\_NUMBER() klasificē vērtībās zem viena definēta parametra ar dažādiem secīgiem cipariem. Funkcijas MIN() un MAX(), kā varētu saprast atgriež mazāko un lielāko vērtību pēc atlasīta parametra, bet ar funkcijas WIDTH\_BUCKET() palīdzību ir iespējams klasificēt vērtības uz “baķetiem” pēc izvēlēta vērtību diapazona.



1. **Vaicājums**

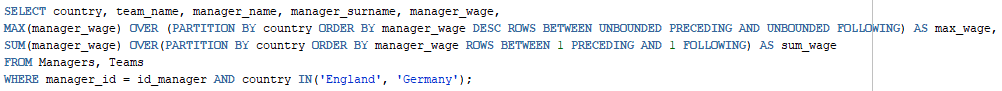
****

Ar funkcijas NTILE() palīdzību ir iespējams sagrupēt datus pēc noteikta parametra ar lietotāju izvēlēto daļu skaitā. Piemērām, grupējot datus ar nosacīto daļu skaitu pēc spēlētāju uzvārda, dati tiks sagrupēti pēc alfabētu secības noteiktajā daļu skaitā.

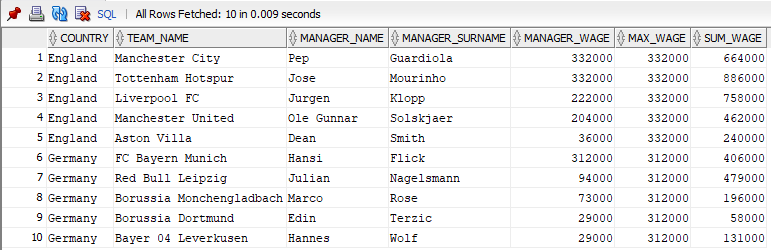
****

## **Vaicājumi ar loga (windowing) funkciju izmantošanu**

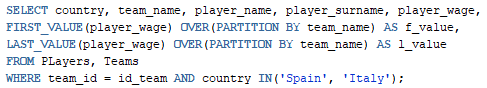
1. **Vaicājums**

****

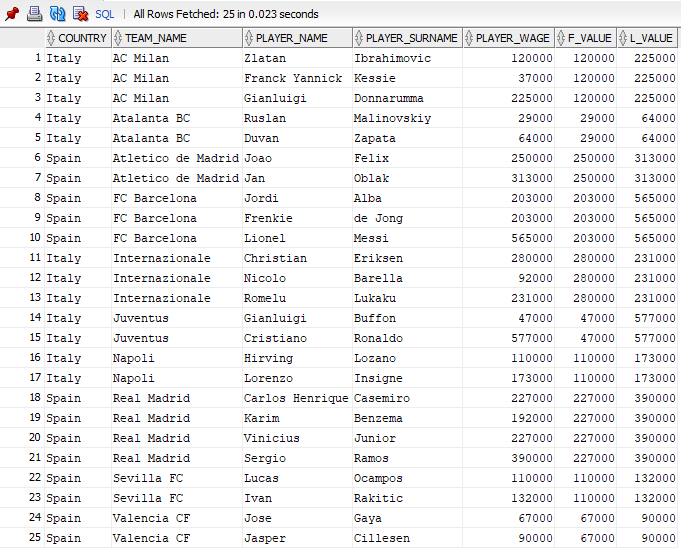
Ar šī vaicājuma palīdzību tiek izvadīta informācija par treneri un viņa komandu. Ar MAX() palīdzību dati tika sagrupēti pēc komandas valsti ar neierobežotu soli treneriem no angļu un vācu komandām. Nākamajā kolonnā tika pārbaudīta ROWS BETWEEN X PRECEDING AND X FOLOWING konstrukcija, kura palīdz grupēt datus pēc izvēlēta soļa.

****

1. **Vaicājums**

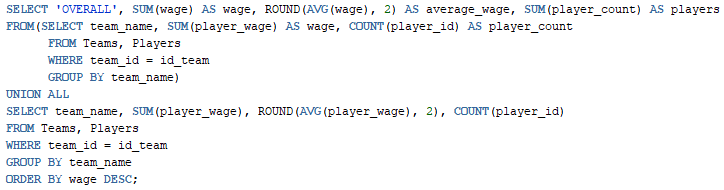
****

Šajā vaicājumā tika pielietotas funkcijas FIRST\_VALUE() un LAST\_VALUE(). Ar OVER(PARTITION BY) konstrukciju un pielietojot augšminētās funkcijas tika izvadīta komandas pirmā spēlētāja atalgojums un pēdējā spēlētāja atalgojums no spāņu un itāļu komandām.

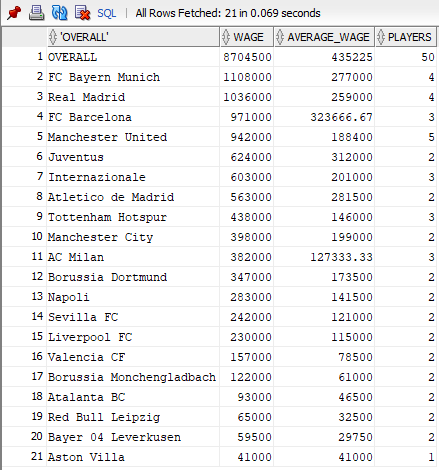
****

## **Vaicājumi ar WITH priekšraksta izmantošanu**

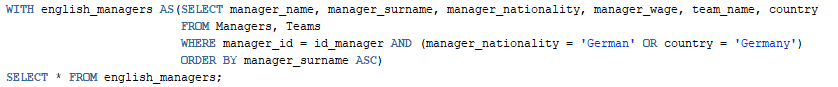
1. **Vaicājums**

****

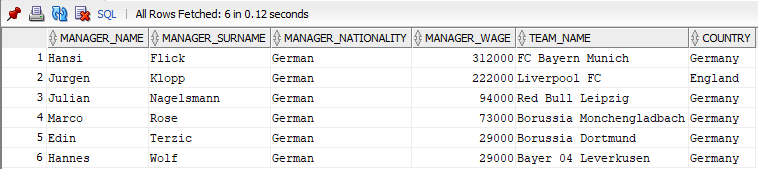
Ar UNION ALL klauzulas palīdzību šajā vaicājumā tika apvienoti divi SELECT vaicājumi, pie nosacījuma, ka šiem abiem vaicājumiem ir vienādas kolonnas. Pirmajā rindā tika izvadīta apkopojoša informācija, bet turpmāk kopēja un vidēja atalgojuma statistika katrai komandai.

****

1. **Vaicājums**

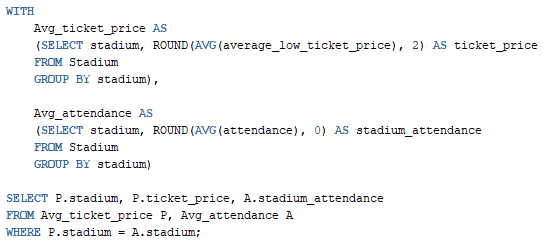
****

Šajā vaicājumā ar WITH klauzulas palīdzību tika atrasti visi treneri no Vācijas vai vācu komandām, izvadot ierakstus alfabētiskā secībā pēc trenera uzvārda.

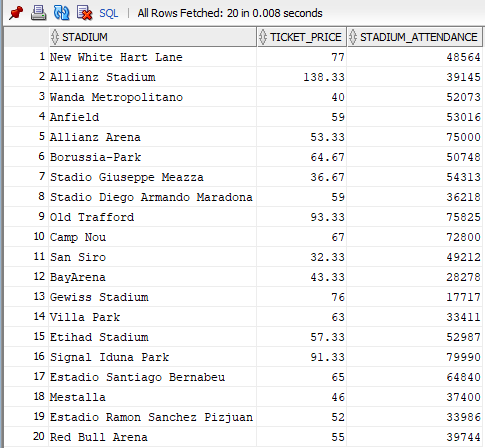
****

## **Rekursīvie vaicājumi ar WITH priekšraksta izmantošanu**

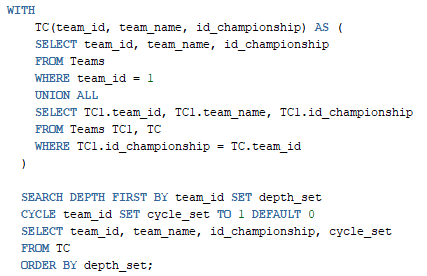
1. **Vaicājums**



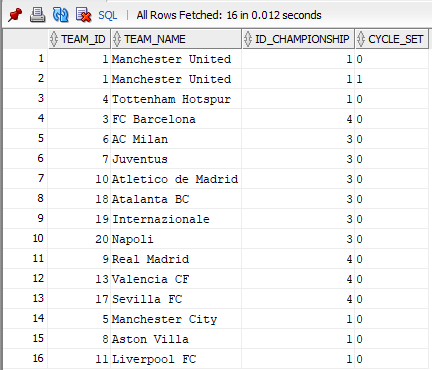
Ar šī vaicājuma palīdzību izmantojot WITH klauzulu tika rekursīvi izvadīti vidēja biļešu cena un vidējais apmeklējums katram stadionam.



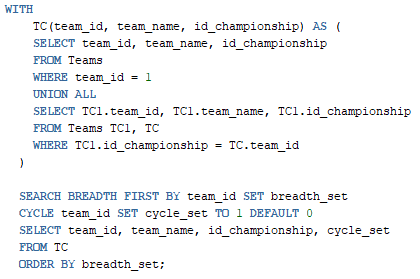
1. **Vaicājums**

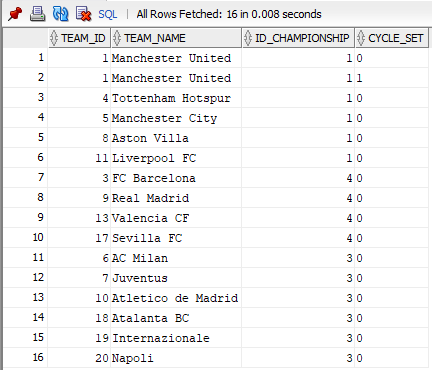
****

Šajā vaicājumā tika pielietota rekursīva konstrukcija, izvadot informāciju par turnīriem un komandām. Tiek izmantota SEARCH DEPTH FIRST BY konstrukcija, un funkcija CYCLE palīdzēja novērst veidojošas cilpas. Nākamajā piemērā arī izpildīju to pašu vaicājumu, tikai ar SEARCH BREAST FIRST BY konstrukciju, rezultāti ir zemāk.

****

* 1. **Vaicājums**

****

****

# **Secinājumi**

Analizējot paveikto darbu ir iespējams secināt, ka šī un iepriekšēja darba izpilde dod lielisko priekšstatu par SQL valodu, darbības pamatprincipiem, vairāku struktūru un konstrukciju pielietojumu. Salīdzinot šo darbu ar iepriekšējo, šīs darbs ir specifiskāks, jo tas aplūko vairāku dažādu vaicājumu tipus un to daudzveidību, bet iepriekšējais darbs bija vairāk apmācošs par SQL valodas sintaksi un darbības pamatiem. Viennozīmīgi balstoties uz to faktu, kā iepriekšējo darbu es arī pildīju izmantojot Oracle SQL Developer rīku un to, ka par pamatu izmantoju iepriekšēja darba datu bāzi, bija vieglāk uzsākts darba uzdevumu izpildi.

Aplūkojot radītas darba gaitā problēmas, kopumā varētu izcelt darbību ar hierarhiskiem un rekursīviem vaicājumiem, kuru izprašana aizņēma vairāku laiku. Lielākā problēma bija cilpu veidošana vaicājumu rezultātos, kas rezultēja ļoti lielu atkārtojošo ierakstu skaitu vai kļūdas paziņojumus. Analizējot vairākus avotus, šīs problēmas tika novērstās, bet kopumā, jā kārtīgi analizēt lekcijas materiālus kombinācijā ar internetā atrodamiem avotiem, tad var iegūt nepieciešamas zināšanas un praktiski pielietot tās uzdevumu izpildē.

Kopumā, šīs darbs aizņēma vairāk laika izpildei nekā iepriekšējais, bet manuprāt, to var uzskatīt par ļoti lietderīgu, jo tas apskata vairākus būtiskus vaicājumu tipus un veiksmīgai uzdevumu izpildei bija jāaplūko daudz informatīvu materiālu, kas viennozīmīgi paplašina zināšanas redzesloku SQL valodā un kopējo izprašanu par datu bāzes koncepciju.

# **Literatūra**

1. Prof. J. Eiduks. Lekciju konspekti priekšmetā “Datu bāzes vadības sistēmas”;
2. estudijas.rtu.lv
3. <https://datubaze.wordpress.com/>
4. <https://docs.oracle.com/cd/B19306_01/server.102/b14200/queries003.htm#:~:text=CONNECT%20BY%20specifies%20the%20relationship,which%20rows%20contain%20the%20loop>.
5. <https://oracle-base.com/articles/misc/rollup-cube-grouping-functions-and-grouping-sets>
6. <https://oracle-base.com/articles/11g/recursive-subquery-factoring-11gr2>
7. <https://www.oracle.com/webfolder/technetwork/tutorials/obe/db/10g/r2/prod/bidw/sqlmodel/sqlmodel_otn.htm>
8. <https://www.w3schools.com/sql/>

# **Pielikums**

## **Darbā izmantotas datu bāzes SQL kods**

DROP TABLE Players;

DROP TABLE Stadium;

DROP TABLE Teams;

DROP TABLE Managers;

DROP TABLE Championship;

CREATE TABLE Managers(

manager\_id number NOT NULL,

manager\_name varchar2(25) NOT NULL,

manager\_surname varchar2(35) NOT NULL,

manager\_nationality varchar2(25),

manager\_wage number CHECK(manager\_wage >= 0),

manager\_age number(2) CHECK(manager\_age >= 0),

manages\_since number(4) CHECK(manages\_since > 0 and manages\_since <= 2021),

CONSTRAINT managers\_pk PRIMARY KEY(manager\_id)

);

CREATE TABLE Championship(

championship\_id number NOT NULL,

championship\_name varchar(50) NOT NULL UNIQUE,

total\_team\_count number(2) NOT NULL CHECK(total\_team\_count > 0),

CONSTRAINT championship\_pk PRIMARY KEY(championship\_id)

);

CREATE TABLE Teams(

team\_id number NOT NULL,

team\_name varchar2(35) NOT NULL UNIQUE,

hometown varchar2(30),

country varchar2(30),

stadium varchar2(40) NOT NULL,

capacity number(6) CHECK(capacity >= 0),

foundation\_year number(4) CHECK (foundation\_year > 0 and foundation\_year <= 2021),

id\_manager number,

id\_championship number,

CONSTRAINT teams\_pk PRIMARY KEY(team\_id),

CONSTRAINT teams\_managers\_fk FOREIGN KEY (id\_manager) REFERENCES Managers (manager\_id),

CONSTRAINT teams\_championship\_fk FOREIGN KEY (id\_championship) REFERENCES Championship (championship\_id)

);

CREATE TABLE Players(

player\_id number NOT NULL,

player\_name varchar2(25) NOT NULL,

player\_surname varchar2(35) NOT NULL,

birthdate date NOT NULL,

position varchar2(30),

player\_nationality varchar2(25),

player\_wage number CHECK(player\_wage >= 0),

at\_club\_since number(4) CHECK (at\_club\_since > 0 AND at\_club\_since <= 2021),

started\_career number(4) CHECK (started\_career > 0 AND started\_career <= 2021),

id\_team number NOT NULL,

CONSTRAINT players\_pk PRIMARY KEY(player\_id),

CONSTRAINT players\_teams\_fk FOREIGN KEY (id\_team) REFERENCES Teams (team\_id)

);

CREATE TABLE Stadium(

record\_id number NOT NULL,

stadium varchar2(40) NOT NULL,

record\_year number(4) CHECK (record\_year > 0 AND record\_year <= 2021),

attendance number(6) CHECK(attendance >= 0),

average\_low\_ticket\_price number(6) CHECK(average\_low\_ticket\_price >= 0),

id\_team number NOT NULL,

CONSTRAINT stadium\_pk PRIMARY KEY(record\_id),

CONSTRAINT stadium\_teams\_fk FOREIGN KEY (id\_team) REFERENCES Teams (team\_id)

);

INSERT INTO Managers VALUES('1', 'Jose', 'Mourinho', 'Portugese', '332000', '58', '2019');

INSERT INTO Managers VALUES('2', 'Andrea', 'Pirlo', 'Italian', '166000', '41', '2020');

INSERT INTO Managers VALUES('3', 'Diego', 'Simeone', 'Argentinian', '464000', '50', '2011');

INSERT INTO Managers VALUES('4', 'Stefano', 'Pioli', 'Italian', '42000', '55', '2019');

INSERT INTO Managers VALUES('5', 'Ole Gunnar', 'Solskjaer', 'Norwegian', '204000', '48', '2018');

INSERT INTO Managers VALUES('6', 'Ronald', 'Koeman', 'Dutch', '215000', '58', '2020');

INSERT INTO Managers VALUES('7', 'Zinadine', 'Zidane', 'French', '232000', '48', '2019');

INSERT INTO Managers VALUES('8', 'Jurgen', 'Klopp', 'German', '222000', '53', '2015');

INSERT INTO Managers VALUES('9', 'Dean', 'Smith', 'English', '36000', '50', '2018');

INSERT INTO Managers VALUES('10', 'Edin', 'Terzic', 'German', '29000', '38', '2020');

INSERT INTO Managers VALUES('11', 'Hansi', 'Flick', 'German', '312000', '56', '2019');

INSERT INTO Managers VALUES('12', 'Pep', 'Guardiola', 'Spanish', '332000', '50', '2016');

INSERT INTO Managers VALUES('13', 'Julen', 'Lopetegui', 'Spanish', '220000', '54', '2019');

INSERT INTO Managers VALUES('14', 'Gian Piero', 'Gasperini', 'Italian', '190000', '63', '2016');

INSERT INTO Managers VALUES('15', 'Antonio', 'Conte', 'Italian', '322000', '51', '2019');

INSERT INTO Managers VALUES('16', 'Marco', 'Rose', 'German', '73000', '44', '2019');

INSERT INTO Managers VALUES('17', 'Hannes', 'Wolf', 'German', '29000', '39', '2021');

INSERT INTO Managers VALUES('18', 'Gennaro', 'Gattuso', 'Italian', '129000', '43', '2019');

INSERT INTO Managers VALUES('19', 'Javi', 'Gracia', 'Spanish', '117000', '50', '2020');

INSERT INTO Managers VALUES('20', 'Julian', 'Nagelsmann', 'German', '94000', '33', '2019');

INSERT INTO Championship VALUES('1', 'Premier League', '20');

INSERT INTO Championship VALUES('2', 'Bundesliga', '18');

INSERT INTO Championship VALUES('3', 'Seria A', '20');

INSERT INTO Championship VALUES('4', 'La Liga', '20');

INSERT INTO Teams VALUES('1', 'Manchester United', 'Manchester', 'England', 'Old Trafford', '76000', '1878', '5', '1');

INSERT INTO Teams VALUES('2', 'Borussia Dortmund', 'Dortmund', 'Germany', 'Signal Iduna Park', '81365', '1909', '10', '2');

INSERT INTO Teams VALUES('3', 'FC Barcelona', 'Barcelona', 'Spain', 'Camp Nou', '99354', '1899', '6', '4');

INSERT INTO Teams VALUES('4', 'Tottenham Hotspur', 'London', 'England', 'New White Hart Lane', '62303', '1882', '1', '1');

INSERT INTO Teams VALUES('5', 'Manchester City', 'Manchester', 'England', 'Etihad Stadium', '55097', '1880', '12', '1');

INSERT INTO Teams VALUES('6', 'AC Milan', 'Milano', 'Italy', 'San Siro', '80018', '1899', '4', '3');

INSERT INTO Teams VALUES('7', 'Juventus', 'Turin', 'Italy', 'Allianz Stadium', '41507', '1897', '2', '3');

INSERT INTO Teams VALUES('8', 'Aston Villa', 'Birmingham', 'England', 'Villa Park', '42785', '1874', '9', '1');

INSERT INTO Teams VALUES('9', 'Real Madrid', 'Madrid', 'Spain', 'Estadio Santiago Bernabeu', '81044', '1902', '7', '4');

INSERT INTO Teams VALUES('10', 'Atletico de Madrid', 'Madrid', 'Spain', 'Wanda Metropolitano', '68456', '1903', '3', '3');

INSERT INTO Teams VALUES('11', 'Liverpool FC', 'Liverpool', 'England', 'Anfield', '53394', '1892', '8', '1');

INSERT INTO Teams VALUES('12', 'FC Bayern Munich', 'Munich', 'Germany', 'Allianz Arena', '75000', '1900', '11', '2');

INSERT INTO Teams VALUES('13', 'Valencia CF', 'Valencia', 'Spain', 'Mestalla', '48600', '1919', '19', '4');

INSERT INTO Teams VALUES('14', 'Red Bull Leipzig', 'Leipzig', 'Germany', 'Red Bull Arena', '42959', '2009', '20', '2');

INSERT INTO Teams VALUES('15', 'Bayer 04 Leverkusen', 'Leverkusen', 'Germany', 'BayArena', '30810', '1904', '17', '2');

INSERT INTO Teams VALUES('16', 'Borussia Monchengladbach', 'Monchengladbach', 'Germany', 'Borussia-Park', '54022', '1900', '16', '2');

INSERT INTO Teams VALUES('17', 'Sevilla FC', 'Seville', 'Spain', 'Estadio Ramon Sanchez Pizjuan', '42714', '1890', '13', '4');

INSERT INTO Teams VALUES('18', 'Atalanta BC', 'Bergamo', 'Italy', 'Gewiss Stadium', '21300', '1907', '14', '3');

INSERT INTO Teams VALUES('19', 'Internazionale', 'Milano', 'Italy', 'Stadio Giuseppe Meazza', '80018', '1908', '15', '3');

INSERT INTO Teams VALUES('20', 'Napoli', 'Naples', 'Italy', 'Stadio Diego Armando Maradona', '54726', '1926', '18', '3');

INSERT INTO Players VALUES('1', 'Cristiano', 'Ronaldo', TO\_DATE('05-02-1985', 'DD-MM-YYYY'), 'Striker', 'Portugese', '577000', '2018', '2002', '7');

INSERT INTO Players VALUES('2', 'Paul', 'Pogba', TO\_DATE('15-03-1993', 'DD-MM-YYYY'), 'Central Midfielder', 'French', '335000', '2016', '2011', '1');

INSERT INTO Players VALUES('3', 'Jack', 'Grealish', TO\_DATE('10-09-1995', 'DD-MM-YYYY'), 'Attacking Midfielder', 'English', '41000', '2001', '2012', '8');

INSERT INTO Players VALUES('4', 'Zlatan', 'Ibrahimovic', TO\_DATE('03-10-1981', 'DD-MM-YYYY'), 'Striker', 'Swedish', '120000', '2020', '1999', '6');

INSERT INTO Players VALUES('5', 'Sergio', 'Ramos', TO\_DATE('30-03-1986', 'DD-MM-YYYY'), 'Center Back', 'Spanish', '390000', '2005', '2003', '9');

INSERT INTO Players VALUES('6', 'Harry', 'Kane', TO\_DATE('28-07-1993', 'DD-MM-YYYY'), 'Striker', 'English', '231000', '2004', '2009', '4');

INSERT INTO Players VALUES('7', 'Jordi', 'Alba', TO\_DATE('21-03-1989', 'DD-MM-YYYY'), 'Left Back', 'Spanish', '203000', '2012', '2007', '3');

INSERT INTO Players VALUES('8', 'Bruno', 'Fernandes', TO\_DATE('08-09-1994', 'DD-MM-YYYY'), 'Attacking Midfielder', 'Portugese', '208000', '2020', '2012', '1');

INSERT INTO Players VALUES('9', 'Jan', 'Oblak', TO\_DATE('07-01-1993', 'DD-MM-YYYY'), 'Goalkeeper', 'Slovenian', '313000', '2014', '2009', '10');

INSERT INTO Players VALUES('10', 'Jadon', 'Sancho', TO\_DATE('25-03-2000', 'DD-MM-YYYY'), 'Right Winger', 'English', '185000', '2017', '2017', '2');

INSERT INTO Players VALUES('11', 'Lorenzo', 'Insigne', TO\_DATE('04-06-1991', 'DD-MM-YYYY'), 'Left Winger', 'Italian', '173000', '2006', '2010', '20');

INSERT INTO Players VALUES('12', 'Sadio', 'Mane', TO\_DATE('10-04-1992', 'DD-MM-YYYY'), 'Left Winger', 'Senegalese', '115000', '2016', '2011', '11');

INSERT INTO Players VALUES('13', 'Thomas', 'Muller', TO\_DATE('13-09-1989', 'DD-MM-YYYY'), 'Attacking Midfielder', 'German', '298000', '2000', '2008', '12');

INSERT INTO Players VALUES('14', 'Romelu', 'Lukaku', TO\_DATE('13-05-1993', 'DD-MM-YYYY'), 'Striker', 'Belgian', '231000', '2019', '2009', '19');

INSERT INTO Players VALUES('15', 'Ibrahima', 'Konate', TO\_DATE('25-05-1999', 'DD-MM-YYYY'), 'Center Back', 'French', '42000', '2017', '2016', '14');

INSERT INTO Players VALUES('16', 'Ruslan', 'Malinovskiy', TO\_DATE('04-05-1993', 'DD-MM-YYYY'), 'Central Midfielder', 'Ukrainian', '29000', '2019', '2011', '18');

INSERT INTO Players VALUES('17', 'Lucas', 'Ocampos', TO\_DATE('11-07-1994', 'DD-MM-YYYY'), 'Left Winger', 'Argentinian', '110000', '2019', '2011', '17');

INSERT INTO Players VALUES('18', 'Sergio', 'Aguero', TO\_DATE('02-06-1988', 'DD-MM-YYYY'), 'Striker', 'Argentinian', '265000', '2011', '2003', '5');

INSERT INTO Players VALUES('19', 'Jasper', 'Cillesen', TO\_DATE('22-04-1989', 'DD-MM-YYYY'), 'Goalkeeper', 'Dutch', '90000', '2019', '2010', '13');

INSERT INTO Players VALUES('20', 'Marcus', 'Thuram', TO\_DATE('06-08-1997', 'DD-MM-YYYY'), 'Striker', 'French', '62000', '2019', '2014', '16');

INSERT INTO Players VALUES('21', 'Timothy', 'Fosu-Mensah', TO\_DATE('02-01-1998', 'DD-MM-YYYY'), 'Right Back', 'Dutch', '16500', '2021', '2016', '15');

INSERT INTO Players VALUES('22', 'Vinicius', 'Junior', TO\_DATE('12-07-2000', 'DD-MM-YYYY'), 'Left Winger', 'Brazilian', '227000', '2018', '2017', '9');

INSERT INTO Players VALUES('23', 'Robert', 'Lewandowski', TO\_DATE('21-08-1988', 'DD-MM-YYYY'), 'Striker', 'Polish', '406000', '2014', '2005', '12');

INSERT INTO Players VALUES('24', 'Frenkie', 'de Jong', TO\_DATE('12-05-1997', 'DD-MM-YYYY'), 'Central Midfielder', 'Dutch', '203000', '2019', '2015', '3');

INSERT INTO Players VALUES('25', 'Franck Yannick', 'Kessie', TO\_DATE('19-12-1996', 'DD-MM-YYYY'), 'Central Midfielder', 'Ivorian', '37000', '2019', '2015', '6');

INSERT INTO Players VALUES('26', 'Hirving', 'Lozano', TO\_DATE('30-07-1995', 'DD-MM-YYYY'), 'Right Winger', 'Mexican', '110000', '2019', '2014', '20');

INSERT INTO Players VALUES('27', 'Toby', 'Alderweireld', TO\_DATE('02-03-1989', 'DD-MM-YYYY'), 'Center Back', 'Belgian', '92000', '2015', '2008', '4');

INSERT INTO Players VALUES('28', 'Erling Braut', 'Haalland', TO\_DATE('21-07-2000', 'DD-MM-YYYY'), 'Striker', 'Norwegian', '162000', '2020', '2015', '2');

INSERT INTO Players VALUES('29', 'Gianluigi', 'Buffon', TO\_DATE('28-01-1978', 'DD-MM-YYYY'), 'Goalkeeper', 'Italian', '47000', '2001', '1995', '7');

INSERT INTO Players VALUES('30', 'Joao', 'Felix', TO\_DATE('10-11-1999', 'DD-MM-YYYY'), 'Attacking Midfielder', 'Portugese', '250000', '2019', '2016', '10');

INSERT INTO Players VALUES('31', 'Leon', 'Goretzka', TO\_DATE('06-02-1995', 'DD-MM-YYYY'), 'Central Midfielder', 'German', '202000', '2018', '2012', '12');

INSERT INTO Players VALUES('32', 'Fabio Henrique', 'Tavares', TO\_DATE('23-10-1993', 'DD-MM-YYYY'), 'Defensive Midfielder', 'Brasilian', '115000', '2018', '2012', '11');

INSERT INTO Players VALUES('33', 'Christian', 'Eriksen', TO\_DATE('14-02-1992', 'DD-MM-YYYY'), 'Attacking Midfielder', 'Dannish', '280000', '2020', '2010', '19');

INSERT INTO Players VALUES('34', 'Marcus', 'Rashford', TO\_DATE('31-10-1997', 'DD-MM-YYYY'), 'Left Winger', 'English', '208000', '2005', '2015', '1');

INSERT INTO Players VALUES('35', 'Ruben', 'Dias', TO\_DATE('14-05-1997', 'DD-MM-YYYY'), 'Center Back', 'Portugese', '133000', '2020', '2015', '5');

INSERT INTO Players VALUES('36', 'Duvan', 'Zapata', TO\_DATE('01-04-1991', 'DD-MM-YYYY'), 'Striker', 'Columbian', '64000', '2018', '2008', '18');

INSERT INTO Players VALUES('37', 'Leon', 'Bailey', TO\_DATE('09-08-1997', 'DD-MM-YYYY'), 'Left Winger', 'Jamaican', '43000', '2017', '2015', '15');

INSERT INTO Players VALUES('38', 'Nicolo', 'Barella', TO\_DATE('07-02-1997', 'DD-MM-YYYY'), 'Central Midfielder', 'Italian', '92000', '2020', '2015', '19');

INSERT INTO Players VALUES('39', 'Gianluigi', 'Donnarumma', TO\_DATE('25-02-1999', 'DD-MM-YYYY'), 'Goalkeeper', 'Italian', '225000', '2013', '2015', '6');

INSERT INTO Players VALUES('40', 'Jose', 'Gaya', TO\_DATE('25-05-1995', 'DD-MM-YYYY'), 'Left Back', 'Spanish', '67000', '2006', '2012', '13');

INSERT INTO Players VALUES('41', 'Carlos Henrique', 'Casemiro', TO\_DATE('23-02-1992', 'DD-MM-YYYY'), 'Defensive Midfielder', 'Brasilian', '227000', '2013', '2010', '9');

INSERT INTO Players VALUES('42', 'Hugo', 'Lloris', TO\_DATE('26-12-1986', 'DD-MM-YYYY'), 'Goalkeeper', 'French', '115000', '2021', '2005', '4');

INSERT INTO Players VALUES('43', 'Mason', 'Greenwood', TO\_DATE('01-10-2001', 'DD-MM-YYYY'), 'Striker', 'English', '87000', '2007', '2018', '1');

INSERT INTO Players VALUES('44', 'Ivan', 'Rakitic', TO\_DATE('10-03-1988', 'DD-MM-YYYY'), 'Central Midfielder', 'Croatian', '132000', '2020', '2005', '17');

INSERT INTO Players VALUES('45', 'Dani', 'Olmo', TO\_DATE('07-05-1998', 'DD-MM-YYYY'), 'Attacking Midfielder', 'Spanish', '23000', '2020', '2015', '14');

INSERT INTO Players VALUES('46', 'Lionel', 'Messi', TO\_DATE('24-06-1987', 'DD-MM-YYYY'), 'Right Winger', 'Argentinian', '565000', '2000', '2003', '3');

INSERT INTO Players VALUES('47', 'Alassane', 'Plea', TO\_DATE('10-03-1993', 'DD-MM-YYYY'), 'Striker', 'French', '60000', '2018', '2010', '16');

INSERT INTO Players VALUES('48', 'Joshua', 'Kimmich', TO\_DATE('08-02-1995', 'DD-MM-YYYY'), 'Defensive Midfielder', 'German', '202000', '2015', '2013', '12');

INSERT INTO Players VALUES('49', 'Karim', 'Benzema', TO\_DATE('19-12-1987', 'DD-MM-YYYY'), 'Striker', 'French', '192000', '2009', '2004', '9');

INSERT INTO Players VALUES('50', 'Aaron', 'Wan-Bissaka', TO\_DATE('26-11-1997', 'DD-MM-YYYY'), 'Right Back', 'English', '104000', '2019', '2016', '1');

INSERT INTO Stadium VALUES('1', 'Old Trafford', '2019', '75498', '99', '1');

INSERT INTO Stadium VALUES('2', 'Old Trafford', '2018', '75976', '87', '1');

INSERT INTO Stadium VALUES('3', 'Old Trafford', '2017', '76000', '94', '1');

INSERT INTO Stadium VALUES('4', 'Signal Iduna Park', '2019', '80820', '99', '2');

INSERT INTO Stadium VALUES('5', 'Signal Iduna Park', '2018', '79496', '92', '2');

INSERT INTO Stadium VALUES('6', 'Signal Iduna Park', '2017', '79653', '83', '2');

INSERT INTO Stadium VALUES('7', 'Camp Nou', '2019', '75208', '68', '3');

INSERT INTO Stadium VALUES('8', 'Camp Nou', '2018', '65731', '76', '3');

INSERT INTO Stadium VALUES('9', 'Camp Nou', '2017', '77461', '57', '3');

INSERT INTO Stadium VALUES('10', 'New White Hart Lane', '2019', '54216', '65', '4');

INSERT INTO Stadium VALUES('11', 'New White Hart Lane', '2018', '59837', '89', '4');

INSERT INTO Stadium VALUES('12', 'New White Hart Lane', '2017', '31639', '77', '4');

INSERT INTO Stadium VALUES('13', 'Etihad Stadium', '2019', '53130', '59', '5');

INSERT INTO Stadium VALUES('14', 'Etihad Stadium', '2018', '52812', '52', '5');

INSERT INTO Stadium VALUES('15', 'Etihad Stadium', '2017', '53019', '61', '5');

INSERT INTO Stadium VALUES('16', 'San Siro', '2019', '54651', '28', '6');

INSERT INTO Stadium VALUES('17', 'San Siro', '2018', '52690', '34', '6');

INSERT INTO Stadium VALUES('18', 'San Siro', '2017', '40294', '35', '6');

INSERT INTO Stadium VALUES('19', 'Allianz Stadium', '2019', '39193', '128', '7');

INSERT INTO Stadium VALUES('20', 'Allianz Stadium', '2018', '38948', '136', '7');

INSERT INTO Stadium VALUES('21', 'Allianz Stadium', '2017', '39294', '151', '7');

INSERT INTO Stadium VALUES('22', 'Villa Park', '2019', '36029', '60', '8');

INSERT INTO Stadium VALUES('23', 'Villa Park', '2018', '32097', '61', '8');

INSERT INTO Stadium VALUES('24', 'Villa Park', '2017', '32107', '68', '8');

INSERT INTO Stadium VALUES('25', 'Estadio Santiago Bernabeu', '2019', '60598', '67', '9');

INSERT INTO Stadium VALUES('26', 'Estadio Santiago Bernabeu', '2018', '65027', '53', '9');

INSERT INTO Stadium VALUES('27', 'Estadio Santiago Bernabeu', '2017', '68896', '75', '9');

INSERT INTO Stadium VALUES('28', 'Wanda Metropolitano', '2019', '56055', '44', '10');

INSERT INTO Stadium VALUES('29', 'Wanda Metropolitano', '2018', '55487', '40', '10');

INSERT INTO Stadium VALUES('30', 'Wanda Metropolitano', '2017', '44678', '36', '10');

INSERT INTO Stadium VALUES('31', 'Anfield', '2019', '52983', '66', '11');

INSERT INTO Stadium VALUES('32', 'Anfield', '2018', '53049', '62', '11');

INSERT INTO Stadium VALUES('33', 'Anfield', '2017', '53016', '49', '11');

INSERT INTO Stadium VALUES('34', 'Allianz Arena', '2019', '75000', '56', '12');

INSERT INTO Stadium VALUES('35', 'Allianz Arena', '2018', '75000', '50', '12');

INSERT INTO Stadium VALUES('36', 'Allianz Arena', '2017', '75000', '54', '12');

INSERT INTO Stadium VALUES('37', 'Mestalla', '2019', '39566', '42', '13');

INSERT INTO Stadium VALUES('38', 'Mestalla', '2018', '38699', '47', '13');

INSERT INTO Stadium VALUES('39', 'Mestalla', '2017', '33936', '49', '13');

INSERT INTO Stadium VALUES('40', 'Red Bull Arena', '2019', '38380', '64', '14');

INSERT INTO Stadium VALUES('41', 'Red Bull Arena', '2018', '39397', '48', '14');

INSERT INTO Stadium VALUES('42', 'Red Bull Arena', '2017', '41454', '53', '14');

INSERT INTO Stadium VALUES('43', 'BayArena', '2019', '27990', '43', '15');

INSERT INTO Stadium VALUES('44', 'BayArena', '2018', '28415', '40', '15');

INSERT INTO Stadium VALUES('45', 'BayArena', '2017', '28428', '47', '15');

INSERT INTO Stadium VALUES('46', 'Borussia-Park', '2019', '49765', '65', '16');

INSERT INTO Stadium VALUES('47', 'Borussia-Park', '2018', '50986', '62', '16');

INSERT INTO Stadium VALUES('48', 'Borussia-Park', '2017', '51494', '67', '16');

INSERT INTO Stadium VALUES('49', 'Estadio Ramon Sanchez Pizjuan', '2019', '36052', '47', '17');

INSERT INTO Stadium VALUES('50', 'Estadio Ramon Sanchez Pizjuan', '2018', '33069', '52', '17');

INSERT INTO Stadium VALUES('51', 'Estadio Ramon Sanchez Pizjuan', '2017', '32837', '57', '17');

INSERT INTO Stadium VALUES('52', 'Gewiss Stadium', '2019', '18284', '80', '18');

INSERT INTO Stadium VALUES('53', 'Gewiss Stadium', '2018', '17921', '64', '18');

INSERT INTO Stadium VALUES('54', 'Gewiss Stadium', '2017', '16946', '84', '18');

INSERT INTO Stadium VALUES('55', 'Stadio Giuseppe Meazza', '2019', '58789', '39', '19');

INSERT INTO Stadium VALUES('56', 'Stadio Giuseppe Meazza', '2018', '57529', '38', '19');

INSERT INTO Stadium VALUES('57', 'Stadio Giuseppe Meazza', '2017', '46620', '33', '19');

INSERT INTO Stadium VALUES('58', 'Stadio Diego Armando Maradona', '2019', '29000', '56', '20');

INSERT INTO Stadium VALUES('59', 'Stadio Diego Armando Maradona', '2018', '43050', '61', '20');

INSERT INTO Stadium VALUES('60', 'Stadio Diego Armando Maradona', '2017', '36605', '60', '20');

SELECT \* FROM Teams;

SELECT \* FROM Managers;

SELECT \* FROM Players;

SELECT \* FROM Stadium;

SELECT \* FROM Championship;

SELECT \* FROM Teams, Managers WHERE manager\_id = id\_manager;

SELECT \* FROM Teams, Managers, Players WHERE team\_id = id\_team AND manager\_id = id\_manager;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1.) Hierarhiskie vaicajumi

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

SELECT player\_id, player\_surname, id\_team

FROM Players

CONNECT BY NOCYCLE PRIOR player\_id = PRIOR id\_team;

SELECT player\_id, player\_surname, id\_team, LEVEL, CONNECT\_BY\_ISCYCLE AS cycle, SYS\_CONNECT\_BY\_PATH(player\_surname, '/') AS path

FROM Players

CONNECT BY NOCYCLE PRIOR player\_id = PRIOR id\_team;

SELECT player\_id, player\_surname, id\_team

FROM Players

START WITH id\_team > 17

CONNECT BY NOCYCLE PRIOR player\_id = PRIOR id\_team

ORDER SIBLINGS BY id\_team;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2.) Faktoru analizes vaicajumi vienai tabulai

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

SELECT record\_year, id\_team, ROUND(AVG(average\_low\_ticket\_price), 2) AS ticket\_price

FROM Stadium

WHERE id\_team BETWEEN 15 AND 20

GROUP BY CUBE(id\_team, record\_year)

ORDER BY id\_team, record\_year;

SELECT record\_year, id\_team, ROUND(AVG(average\_low\_ticket\_price), 2) AS ticket\_price

FROM Stadium

WHERE id\_team BETWEEN 15 AND 20

GROUP BY ROLLUP(id\_team, record\_year)

ORDER BY id\_team, record\_year;

SELECT record\_year, id\_team, ROUND(AVG(attendance), 0) AS attendance,

GROUPING(id\_team) AS g\_id,

GROUPING(record\_year) AS g\_ry

FROM Stadium

WHERE (id\_team BETWEEN 5 AND 10) AND record\_year IN ('2017', '2018')

GROUP BY CUBE(id\_team, record\_year)

ORDER BY id\_team, record\_year;

SELECT record\_year, id\_team, ROUND(AVG(attendance), 0) AS attendance,

GROUPING(id\_team) AS g\_id,

GROUPING(record\_year) AS g\_ry

FROM Stadium

WHERE id\_team BETWEEN 5 AND 10

GROUP BY CUBE(id\_team, record\_year)

HAVING GROUPING(id\_team) = 1 OR GROUPING(record\_year) = 1

ORDER BY id\_team, record\_year;

SELECT record\_year, id\_team, ROUND(AVG(average\_low\_ticket\_price), 2) AS ticket\_price,

GROUPING\_ID(record\_year, id\_team) AS grouping\_id,

GROUP\_ID() AS group\_id

FROM Stadium

WHERE record\_year IN('2018', '2019')

GROUP BY GROUPING SETS(record\_year, CUBE(record\_year, id\_team))

ORDER BY record\_year, id\_team;

SELECT record\_year, id\_team, ROUND(AVG(average\_low\_ticket\_price), 2) AS ticket\_price,

GROUPING\_ID(record\_year, id\_team) AS grouping\_id,

GROUP\_ID() AS group\_id

FROM Stadium

WHERE record\_year IN('2018', '2019')

GROUP BY GROUPING SETS(record\_year, id\_team)

ORDER BY record\_year, id\_team;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

3.) Vaicajumi tabulu zvaigznes shemai

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

SELECT championship\_id, COUNT(DISTINCT team\_id) AS team\_count, position, SUM(player\_wage) AS wage,

GROUPING(position) AS g\_pos

FROM Teams, Championship, Players

WHERE championship\_id = id\_championship AND team\_id = id\_team AND championship\_id IN ('1', '2')

GROUP BY CUBE(championship\_id, position)

ORDER BY g\_pos, championship\_id;

SELECT championship\_id, team\_name, position, SUM(player\_wage) AS wage,

GROUPING(position) AS g\_pos

FROM Teams, Championship, Players

WHERE championship\_id = id\_championship AND team\_id = id\_team AND championship\_id IN ('3', '4')

GROUP BY GROUPING SETS(championship\_id, team\_name, position);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

4.) Virtualas daudzdimensiju tabulas vaicajumi ar SQL Model klauzulas izmantosanu

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

SELECT stadium, record\_year, id\_team, attendance

FROM Stadium

MODEL PARTITION BY(stadium) DIMENSION BY(record\_year, id\_team)

MEASURES(attendance)

RULES();

SELECT stadium, record\_year, id\_team, ROUND(attendance, 0) AS attendance

FROM Stadium

WHERE id\_team = 2

MODEL PARTITION BY(stadium) DIMENSION BY(record\_year, id\_team)

MEASURES(attendance)

RULES(attendance['2020', 2] = AVG(attendance)[record\_year BETWEEN 2017 and 2019, 2]);

SELECT stadium, record\_year, id\_team, attendance

FROM Stadium

WHERE id\_team = 2

MODEL PARTITION BY(stadium) DIMENSION BY(record\_year, id\_team)

MEASURES(attendance)

RULES(UPDATE attendance['2019', 2] = 80000);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

5.) Vaicajumi ar analizes funkciju izmantosanu

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

SELECT id\_team, player\_name, player\_surname, player\_wage,

ROUND(player\_wage\*100/SUM(player\_wage) OVER(PARTITION BY id\_team), 2) AS wage\_percent,

RANK() OVER(PARTITION BY id\_team ORDER BY player\_wage) AS rank,

DENSE\_RANK() OVER(PARTITION BY id\_team ORDER BY player\_wage) AS d\_rank,

ROW\_NUMBER() OVER(PARTITION BY id\_team ORDER BY player\_wage) AS r\_number,

MIN(player\_wage) KEEP(DENSE\_RANK FIRST ORDER BY player\_wage) OVER(PARTITION BY id\_team) AS min\_wage,

MAX(player\_wage) KEEP(DENSE\_RANK LAST ORDER BY player\_wage) OVER(PARTITION BY id\_team) AS max\_wage,

WIDTH\_BUCKET(player\_wage, 80000, 600000, 5) AS width\_bucket

FROM Players

WHERE id\_team <= 5;

SELECT id\_team, player\_name, player\_surname, player\_wage,

NTILE(5) OVER(ORDER BY player\_surname) AS group\_num

FROM Players

WHERE id\_team <= 5;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

6.) Vaicajumi ar loga (windowing) funkciju izmantosanu

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

SELECT country, team\_name, manager\_name, manager\_surname, manager\_wage,

MAX(manager\_wage) OVER (PARTITION BY country ORDER BY manager\_wage DESC ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) AS max\_wage,

SUM(manager\_wage) OVER(PARTITION BY country ORDER BY manager\_wage ROWS BETWEEN 1 PRECEDING AND 1 FOLLOWING) AS sum\_wage

FROM Managers, Teams

WHERE manager\_id = id\_manager AND country IN('England', 'Germany');

SELECT country, team\_name, player\_name, player\_surname, player\_wage,

FIRST\_VALUE(player\_wage) OVER(PARTITION BY team\_name) AS f\_value,

LAST\_VALUE(player\_wage) OVER(PARTITION BY team\_name) AS l\_value

FROM PLayers, Teams

WHERE team\_id = id\_team AND country IN('Spain', 'Italy');

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

7.) Vaicajumi ar WITH prieksraksta izmantosanu

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

SELECT 'OVERALL', SUM(wage) AS wage, ROUND(AVG(wage), 2) AS average\_wage, SUM(player\_count) AS players

FROM(SELECT team\_name, SUM(player\_wage) AS wage, COUNT(player\_id) AS player\_count

FROM Teams, Players

WHERE team\_id = id\_team

GROUP BY team\_name)

UNION ALL

SELECT team\_name, SUM(player\_wage), ROUND(AVG(player\_wage), 2), COUNT(player\_id)

FROM Teams, Players

WHERE team\_id = id\_team

GROUP BY team\_name

ORDER BY wage DESC;

WITH english\_managers AS(SELECT manager\_name, manager\_surname, manager\_nationality, manager\_wage, team\_name, country

FROM Managers, Teams

WHERE manager\_id = id\_manager AND (manager\_nationality = 'German' OR country = 'Germany')

ORDER BY manager\_surname ASC)

SELECT \* FROM english\_managers;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

8.) Rekursivie vaicajumi ar WITH prieksraksta izmantosanu

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

WITH

Avg\_ticket\_price AS

(SELECT stadium, ROUND(AVG(average\_low\_ticket\_price), 2) AS ticket\_price

FROM Stadium

GROUP BY stadium),

Avg\_attendance AS

(SELECT stadium, ROUND(AVG(attendance), 0) AS stadium\_attendance

FROM Stadium

GROUP BY stadium)

SELECT P.stadium, P.ticket\_price, A.stadium\_attendance

FROM Avg\_ticket\_price P, Avg\_attendance A

WHERE P.stadium = A.stadium;

WITH

TC(team\_id, team\_name, id\_championship) AS (

SELECT team\_id, team\_name, id\_championship

FROM Teams

WHERE team\_id = 1

UNION ALL

SELECT TC1.team\_id, TC1.team\_name, TC1.id\_championship

FROM Teams TC1, TC

WHERE TC1.id\_championship = TC.team\_id

)

SEARCH DEPTH FIRST BY team\_id SET depth\_set

CYCLE team\_id SET cycle\_set TO 1 DEFAULT 0

SELECT team\_id, team\_name, id\_championship, cycle\_set

FROM TC

ORDER BY depth\_set;

WITH

TC(team\_id, team\_name, id\_championship) AS (

SELECT team\_id, team\_name, id\_championship

FROM Teams

WHERE team\_id = 1

UNION ALL

SELECT TC1.team\_id, TC1.team\_name, TC1.id\_championship

FROM Teams TC1, TC

WHERE TC1.id\_championship = TC.team\_id

)

SEARCH BREADTH FIRST BY team\_id SET breadth\_set

CYCLE team\_id SET cycle\_set TO 1 DEFAULT 0

SELECT team\_id, team\_name, id\_championship, cycle\_set

FROM TC

ORDER BY breadth\_set;