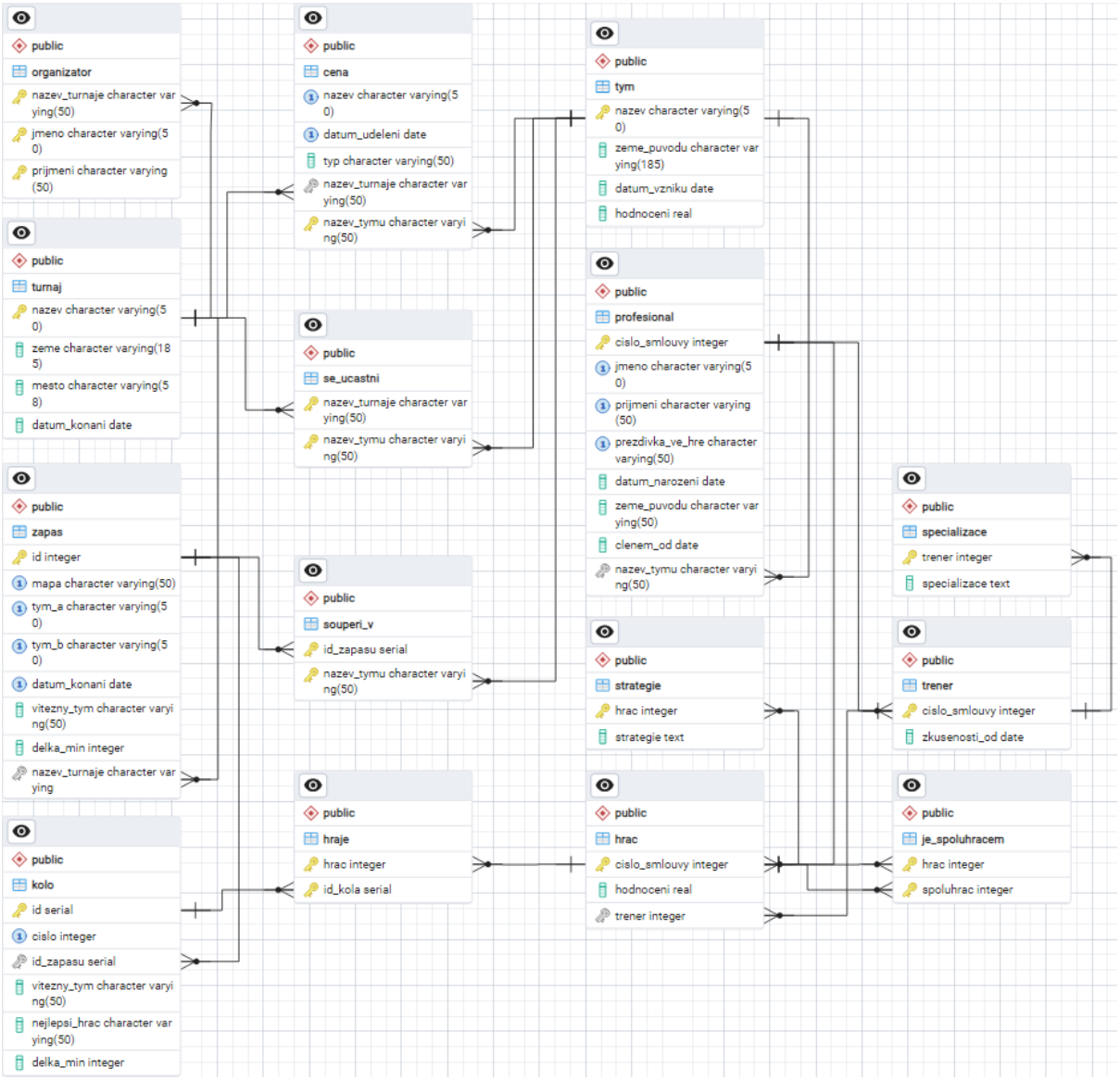
Turnaj v počítačové hře - CP3 DBS

Na začátek přidávám obrázek ER a relačního modelu semestrální práce podle zadání.

ER model:



Relační model databáze turnaje v počítačové hře

Turnaj(nazev, mesto, zeme, datum_konani)

Organizator(nazev_turnaje, organizator)

FK: (nazev_turnaje) ⊆ Turnaj(nazev)

Cena(nazev, datum_udeleni, typ, nazev_turnaje, nazev_tymu)

FK: (nazev_turnaje) ⊆ Turnaj(nazev)

FK: (nazev_tymu) ⊆ Tym(nazev)

Zapas(mapa, tym_A, tym_B, datum_konani, vitezny_tym, delka, nazev_turnaje)

FK: (nazev_turnaje) ⊆ Turnaj(nazev)

Tym(nazev, zeme_puvodu, datum_vzniku, hodnoceni)

Se_ucastni(nazev_turnaje, nazev_tymu)

FK: (nazev_turnaje) ⊆ Turnaj(nazev)

FK: (nazev_tymu) ⊆ Tym(nazev)

Souperi_v(mapa_zapasu, tym_A_zapasu, tym_B_zapasu, datum_konani_zapasu, nazev_tymu)

FK: (mapa_zapasu, tym_A_zapasu, tym_B_zapasu, datum_konani_zapasu) ⊆ Zapas(mapa, tym_A, tym_B, datum_konani)

FK: (nazev_tymu) ⊆ Tym(nazev)

Kolo(cislo, mapa_zapasu, tym_A_zapasu, tym_B_zapasu, datum_konani_zapasu, vitezny_tym, nejlepsi_hrac, minuty, sekundy)

FK: (mapa_zapasu, tym_A_zapasu, tym_B_zapasu, datum_konani_zapasu) ⊆ Zapas(mapa, tym_A, tym_B, datum_konani)

Profesional(cislo_smlouvy, jmeno, prezdivka_ve_hre, datum_narozeni, zeme_puvodu, clenem_od, nazev_tymu)

FK: (nazev_tymu) ⊆ Tym(nazev)

Hrac(cislo_smlouvy, hodnoceni, trener)

FK: (cislo_smlouvy) ⊆ Profesional(cislo_smlouvy)

FK: (trener) ⊆ Trener(cislo_smlouvy)

Strategie(hrac, strategie)

FK: (hrac) ⊆ Hrac(cislo_smlouvy)

Je_spoluhracem(hrac, spoluhrac)

FK: (hrac) ⊆ Hrac(cislo_smlouvy)

FK: (spoluhrac) ⊆ Hrac(cislo_smlouvy)

Hraje(hrac, cislo_kola, mapa_zapasu, tym_A_zapasu, tym_B_zapasu, datum_konani_zapasu)

FK: (hrac) ⊆ Hrac(cislo_smlouvy))

FK: (cislo_kola, mapa_zapasu, tym_A_zapasu, tym_B_zapasu, datum_konani_zapasu) ⊆ Kolo(cislo, mapa_zapasu, tym_A_zapasu, tym_B_zapasu, datum_konani_zapasu)

Trener(cislo_smlouvy, zkusenosti_od)

FK: (cislo_smlouvy) ⊆ Profesional(cislo_smlouvy))

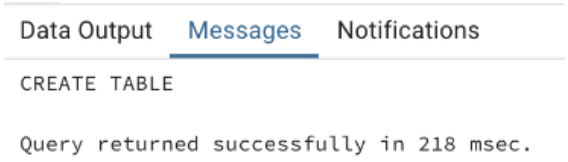
Specializace(trener, specializace)

FK: (trener) ⊆ Trener(cislo_smlouvy)

Nejdříve uvádím SQL příkazy na vytvoření databáze. Přikládám obrázky dotazů i výsledek dotazu jeden po druhém.

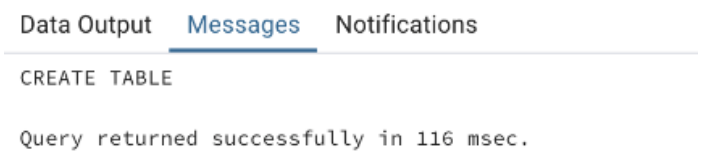
1. Dotaz na vytvoření tabulky Turnaj a výsledek dotazu v Postgresql.

```
CREATE TABLE Turnaj
(
    nazev varchar(50) NOT NULL,
    zeme varchar(185) NOT NULL,
    mesto varchar(58) NOT NULL,
    datum_konani date NOT NULL,
    PRIMARY KEY (nazev)
);
```



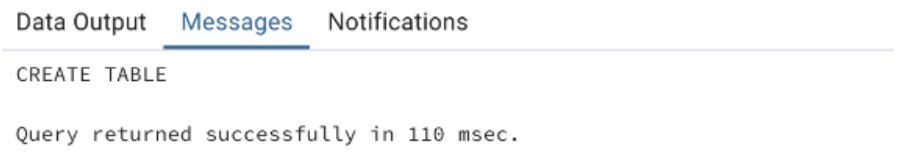
2. Dotaz na vytvoření tabulky Tym a výsledek dotazu v Postgresql.

```
CREATE TABLE Tym
(
    nazev varchar(50) NOT NULL,
    zeme_puvodu varchar(185) NOT NULL,
    datum_vzniku date NOT NULL,
    hodnoceni float4 NOT NULL,
    PRIMARY KEY (nazev)
);
```



3. Dotaz na vytvoření tabulky Cena a výsledek dotazu v Postgresql. Obsahuje referenci na tabulky Turnaj a Tym. U obou jsou použita direktiva ON UPDATE CASCADE, protože při změně cizího klíče (Turnaj či Tym) je nutné změnit i referenci na něj. Direktiva ON DELETE RESTRICT zajišťuje, že při pokusu vymazat Turnaj či Tym, na který reference odkazuje, bude tento pokus zamítnut. Při získání ceny týmem v turnaji, již není možné tým či turnaj vymazat.

```
CREATE TABLE Cena
(
        nazev varchar(50) NOT NULL,
        datum_udeleni date NOT NULL,
        typ varchar(50) NOT NULL,
        nazev_turnaje varchar(50) NOT NULL,
        nazev_tymu varchar(50) NOT NULL,
        PRIMARY KEY (nazev_tymu),
        UNIQUE (nazev, datum_udeleni),
        FOREIGN KEY (nazev_turnaje) REFERENCES Turnaj(nazev)
                ON UPDATE CASCADE
                ON DELETE RESTRICT,
        FOREIGN KEY (nazev_tymu) REFERENCES Tym(nazev)
                ON UPDATE CASCADE
                ON DELETE RESTRICT
);
```



4. Dotaz na vytvoření tabulky Organizator a výsledek dotazu v Postgresql. Cizím klíčem je reference na Turnaj. Opět jako u tabulky Cena jsou obě direktivy nastaveny stejně ze stejných důvodů.

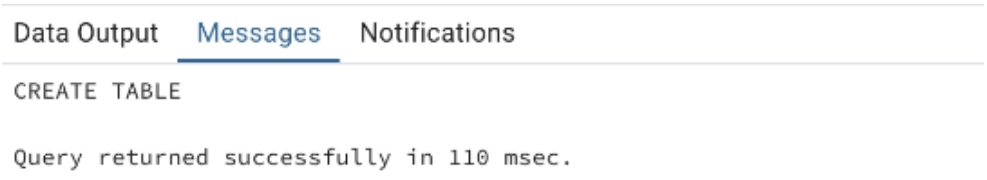
```
CREATE TABLE Organizator
(
        nazev_turnaje varchar(50) NOT NULL,
        jmeno varchar(50) NOT NULL,
        prijmeni varchar(50) NOT NULL,
        PRIMARY KEY (nazev_turnaje, jmeno, prijmeni),
        FOREIGN KEY (nazev_turnaje) REFERENCES Turnaj(nazev)
                ON UPDATE CASCADE
                ON DELETE RESTRICT
);
```

5. Dotaz na vytvoření tabulky Zapas a výsledek dotazu v Postgresql. Tato tabulka má vytvořený umělý autoinkrementovaný klíč id z důvodu zjednodušení dotazů a referencí na tento klíč/tabulku. Nahrazený složený klíč je zde UNIQUE, unikátní. Cizím klíčem je reference na Turnaj. Direktivy jsou nastaveny jako u předchozích dvou tabulek Cena a Organizator.

```
CREATE TABLE Zapas
(
        id serial NOT NULL,
        mapa varchar(50) NOT NULL,
        tym_A varchar(50) NOT NULL,
        tym_B varchar(50) NOT NULL,
        datum_konani date NOT NULL,
        vitezny_tym varchar(50) NOT NULL,
        delka int4 NOT NULL,
        nazev_turnaje varchar(50) NOT NULL,
        PRIMARY KEY (id),
        UNIQUE (mapa, tym_A, tym_B, datum_konani),
        FOREIGN KEY (nazev_turnaje) REFERENCES Turnaj(nazev)
                ON UPDATE CASCADE
                ON DELETE RESTRICT
);
```

6. Dotaz na vytvoření tabulky Se_ucastni a výsledek dotazu v Postgresql. Tato tabulka je použita pro propojení tabulek Turnaj a Tym. Proto používá u obou direktiv CASCADE, protože při změně i vymazání u tabulek Turnaj či Tym musí dojít ke změně či vymazání i propojení zde.

```
CREATE TABLE Se_ucastni
(
nazev_turnaje varchar(50) NOT NULL,
nazev_tymu varchar(50) NOT NULL,
PRIMARY KEY (nazev_turnaje, nazev_tymu),
FOREIGN KEY (nazev_turnaje) REFERENCES Turnaj(nazev)
        ON UPDATE CASCADE
        ON DELETE CASCADE,
FOREIGN KEY (nazev_tymu) REFERENCES Tym(nazev)
        ON UPDATE CASCADE
        ON DELETE CASCADE
);
```

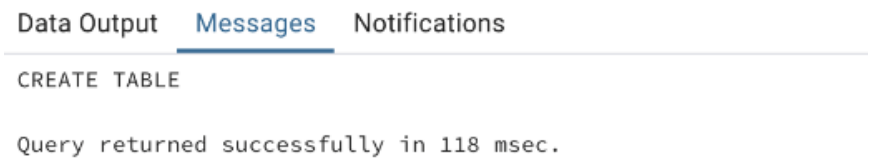


7. Dotaz na vytvoření tabulky Souperi_v a výsledek dotazu v Postgresql. Využívá referenci na tabulku Tym a obě direktivy jsou použity stejně jako u tabulky Se_ucastni.

```
CREATE TABLE Souperi_v
(
id_zapasu serial NOT NULL,
nazev_tymu varchar(50) NOT NULL,
PRIMARY KEY (id_zapasu, nazev_tymu),
FOREIGN KEY (id_zapasu) REFERENCES Zapas(id)
        ON UPDATE CASCADE
        ON DELETE CASCADE,
FOREIGN KEY (nazev, tymu) REFERENCES Tym(nazev)
        ON UPDATE CASCADE
        ON DELETE CASCADE
);
```

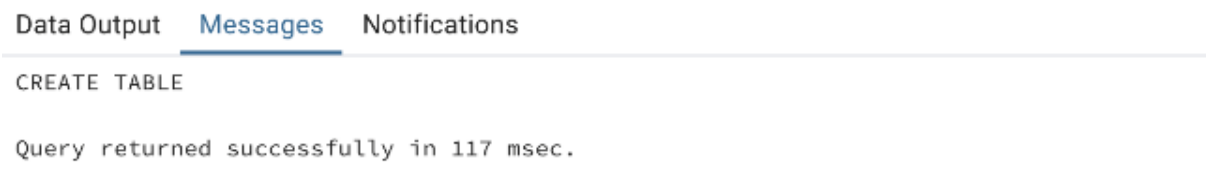
8. Dotaz na vytvoření tabulky Profesional a výsledek dotazu v Postgresql. Referencí je tabulka Tym. U obou direktiv je použito CASCADE, protože při vymazání týmu z databáze se vymažou i členové tohoto týmu. Při změně v tabulce Tym se změní i reference v této tabulce.

```
CREATE TABLE Profesional
(
        cislo_smlouvy int4 NOT NULL,
        jmeno varchar(50) NOT NULL,
        prijmeni varchar(50) NOT NULL,
        prezdivka_ve_hre varchar(50) NOT NULL,
        datum_konani date NOT NULL,
        zeme_puvodu varchar(50) NOT NULL,
        clenem_od date NOT NULL,
        nazev_tymu varchar(50) NOT NULL,
        PRIMARY KEY (cislo_smlouvy),
        UNIQUE (jmeno, prijmeni, prezdivka_ve_hre),
        FOREIGN KEY (nazev_tymu) REFERENCES Tym(nazev)
                ON UPDATE CASCADE
                ON DELETE CASCADE
);
```



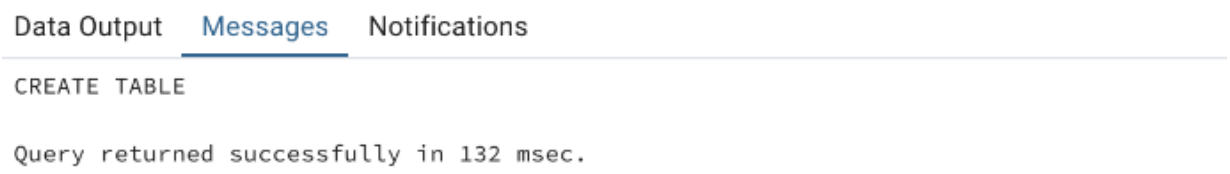
9. Dotaz na vytvoření tabulky Trener a výsledek dotazu v Postgresql. Cizím klíčem je reference na tabulku Profesional. Tabulka vlastně dědí od tabulky Profesional a přidává další atributy. Proto při změně či vymazání je použito CASCADE u obou direktiv.

```
CREATE TABLE Trener
(
        cislo_smlouvy int4 NOT NULL,
        zkusenosti_od date NOT NULL,
        PRIMARY KEY (cislo_smlouvy),
        FOREIGN KEY (cislo_smlouvy) REFERENCES Profesional(cislo_smlouvy)
                ON UPDATE CASCADE
                ON DELETE CASCADE
);
```



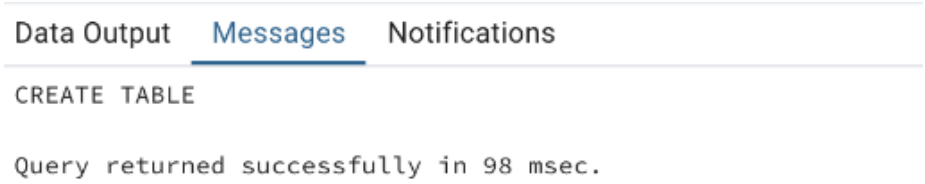
10. Dotaz na vytvoření tabulky Hrac a výsledek dotazu v Postgresql. Referencí je tabulka Profesional a Trener. Funguje prakticky stejně jako tabulka Trener. U obou cizích klíčů je opět použito CASCADE ze stejných důvodů jako u tabulky Trener.

```
CREATE TABLE Hrac
(
        cislo_smlouvy int4 NOT NULL,
        hodnoceni float4 NOT NULL,
        trener int4 NOT NULL,
        PRIMARY KEY (cislo_smlouvy),
        FOREIGN KEY (cislo_smlouvy) REFERENCES Profesional(cislo_smlouvy)
                ON UPDATE CASCADE
                ON DELETE CASCADE,
        FOREIGN KEY (trener) REFERENCES Trener(cislo_smlouvy)
                ON UPDATE CASCADE
                ON DELETE CASCADE
);
```



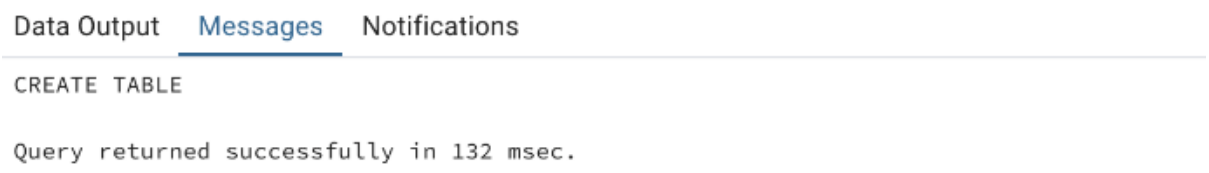
11. Dotaz na vytvoření tabulky Kolo a výsledek dotazu v Postgresql. Zde je vytvořen umělý autoinkrementovaný identifikátor id z důvodu propojení s jinými tabulkami. Minulý složený identifikátor je nyní UNIQUE, unikátní. Použití u direktiv referenčního klíče má stejný důvod jako u tabulky Profesional.

```
CREATE TABLE Kolo
(
        id serial NOT NULL,
        cislo int4 NOT NULL,
        id_zapasu serial NOT NULL,
        vitezny_tym varchar(50) NOT NULL,
        nejlepsi_hrac varchar(50) NOT NULL,
        delka int4 NOT NULL,
        PRIMARY KEY (id),
        UNIQUE (cislo, id_zapasu),
        FOREIGN KEY (id_zapasu) REFERENCES Zapas(id)
                ON UPDATE CASCADE
                ON DELETE CASCADE
);
```



12. Dotaz na vytvoření tabulky Hraje a výsledek dotazu v Postgresql. Tato tabulka funguje jako propojení tabulek Hrac a Kolo. Proto jsou direktivy použity stejně jako u Se_ucastni nebo Souperi_v. Hlavně kvůli této tabulce se musel v tabulce Kolo založit umělý klíč.

```
CREATE TABLE Hraje
(
        hrac int4 NOT NULL,
        id_kolo seriál NOT NULL,
        PRIMARY KEY (hrac, id_kola),
        FOREIGN KEY (hrac) REFERENCES Hrac(cislo_smlouvy)
                ON UPDATE CASCADE
                ON DELETE CASCADE,
        FOREIGN KEY (id_kola) REFERENCES Kolo(id)
                ON UPDATE CASCADE
                ON DELETE CASCADE
);
```



13. Dotaz na vytvoření tabulky Specializace a výsledek dotazu v Postgresql. Tabulka Trener je cizím klíčem. Opět direktivy jsou použity stejně jako u tabulek Trener či Hrac atd.

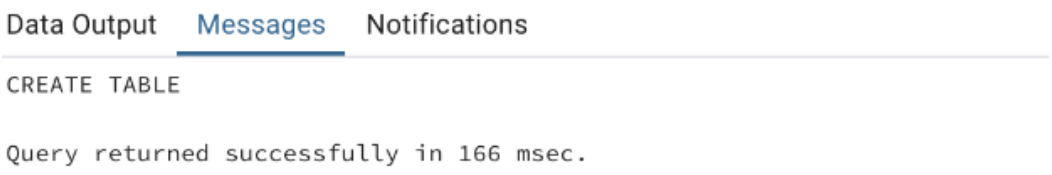
```
CREATE TABLE Specializace
(
        trener int4 NOT NULL,
        specializace text NOT NULL,
        PRIMARY KEY (trener),
        FOREIGN KEY (trener) REFERENCES Trener(cislo_smlouvy)
                ON UPDATE CASCADE
                ON DELETE CASCADE
);
```

14. Dotaz na vytvoření tabulky Strategie a výsledek dotazu v Postgresql. Funguje stejně jako tabulka Specializace akorát pro tabulku Hrac, na kterou má referenci. Direktivy u cizího klíče jsou použity stejně jako u tabulky Specializace.

```
CREATE TABLE Strategie
(
        hrac int4 NOT NULL,
        strategie text NOT NULL,
        PRIMARY KEY (hrac),
        FOREIGN KEY (hrac) REFERENCES Hrac(cislo_smlouvy)
                ON UPDATE CASCADE
                ON DELETE CASCADE
);
```

15. Dotaz na vytvoření tabulky Je_spoluhracem a výsledek dotazu v Postgresql. Propojuje každého hráče s jeho spoluhráčem v týmu. Z tohoto důvodu je použito u direktiv obou cizích klíčů CASCADE, protože při změně/vymazání hráče se provede změna/vymazání i zde v tabulce.

```
CREATE TABLE Je_spoluhracem
(
        hrac int4 NOT NULL,
        spoluhrac int4 NOT NULL,
        PRIMARY KEY (hrac, spoluhrac),
        FOREIGN KEY (hrac) REFERENCES Hrac(cislo_smlouvy)
                ON UPDATE CASCADE
                ON DELETE CASCADE,
        FOREIGN KEY (spoluhrac) REFERENCES Hrac(cislo_smlouvy)
                ON UPDATE CASCADE
                ON DELETE CASCADE
);
```

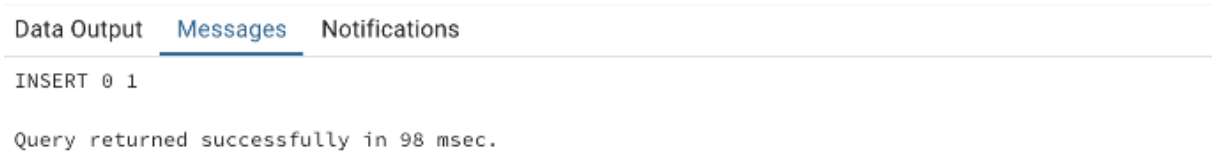


Přidávám obrázky dotazů SQL ohledně insertu do zmíněných tabulek a výsledky dotazů v Postgresql. Dotazy jsou postupně pro tabulky Turnaj, Tym, Cena, Organizator, Se_ucastni, Zapas, Profesional, Souperi_v, Trener, Hrac, Je_spoluhracem, Specializace, Strategie, Kolo, Hraje.

INSERT INTO turnaj VALUES ('ESL Majors Series One', 'Polsko', 'Katowice', '2014-7-1');

INSERT INTO tym VALUES ('Virtus Pro', 'Polsko', '2012-5-5', '4.5');

INSERT INTO tym VALUES ('Natus Vincere', 'Ruská Federace', '2012-3-10', '4.6');



INSERT INTO cena VALUES (‚Katowice Masters 2014', ‚2014-8-10', ‚zlaty pohar',

‚ESL Majors Series One', ‚Virtus Pro');

INSERT INTO organizátor VALUES (‚ESL Majors Series One', ‚Tadeusz', ‚Kozlowski');

INSERT INTO organizátor VALUES (‚ESL Majors Series One', ‚Michal', ‚Wazowski');

INSERT INTO organizátor VALUES (‚ESL Majors Series One', ‚Kyle', ‚Broflovski');

INSERT INTO se_ucastni VALUES (‚ESL Majors Series One', ‚Virtus Pro');

INSERT INTO se_ucastni VALUES (‚ESL Majors Series One', ‚Natus Vincere');

Zbytek dotazů ohledně vkládání přikládám v obrázcích.

```
INSERT INTO zapas (mapa, tym_a, tym_b, datum_konani, vitezny_tym, delka, nazev_turnaje)
    VALUES ('Inferno', 'Virtus Pro', 'Natus Vincere', '2014-7-2', 'Virtus Pro',
        '50', 'ESL Majors Series One');
INSERT INTO zapas (mapa, tym_a, tym_b, datum_konani, vitezny_tym, delka, nazev_turnaje)
    VALUES ('Mirage', 'Virtus Pro', 'Natus Vincere', '2014-7-5', 'Natus Vincere',
        '45', 'ESL Majors Series One');
INSERT INTO zapas (mapa, tym_a, tym_b, datum_konani, vitezny_tym, delka, nazev_turnaje)
    VALUES ('Nuke', 'Virtus Pro', 'Natus Vincere', '2014-7-7', 'Virtus Pro',
        '55', 'ESL Majors Series One');
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 106 msec.
```

```
INSERT INTO profesional VALUES ('12345678', 'Jan', 'Kowalski', 'kawasaki',
                                '1992-3-21', 'Polsko', '2012-5-5', 'Virtus Pro');
INSERT INTO profesional VALUES ('54637825', 'Wojciech', 'Nowak', 'nowa',
                                '1994-4-30', 'Polsko', '2012-5-5', 'Virtus Pro');
INSERT INTO profesional VALUES ('85736409', 'Mateusz', 'Zieliński', 'zeli',
                                '1993-8-4', 'Polsko', '2012-5-5', 'Virtus Pro');
INSERT INTO profesional VALUES ('74659023', 'Jakub', 'Wójcik', 'wojtek',
                                '1992-12-1', 'Polsko', '2012-5-5', 'Virtus Pro');
INSERT INTO profesional VALUES ('46395712', 'Tomasz', 'Dąbrowski', 'dabrowski',
                                '1994-7-16', 'Polsko', '2012-5-5', 'Virtus Pro');
INSERT INTO profesional VALUES ('55555555', 'Dmitry', 'Ivanov', 'hey_ivan',
                                '1994-8-9', 'Ruská Federace', '2012-3-10', 'Natus Vincere');
INSERT INTO profesional VALUES ('66774488', 'Ivan', 'Petrov', 'petra',
                                '1991-3-6', 'Ruská Federace', '2012-3-10', 'Natus Vincere');
INSERT INTO profesional VALUES ('88337755', 'Alexander', 'Kuznetsov', 'kuznyak',
                                '1992-11-19', 'Ruská Federace', '2012-3-10', 'Natus Vincere');
INSERT INTO profesional VALUES ('12345432', 'Maxim', 'Sokolov', 'sokol',
                                '1993-10-3', 'Ruská Federace', '2012-3-10', 'Natus Vincere');
INSERT INTO profesional VALUES ('87504738', 'Sergey', 'Popov', 'popovski',
                                '1991-3-22', 'Ruská Federace', '2012-3-10', 'Natus Vincere');
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 110 msec.
```

```
INSERT INTO souperi_v VALUES ('1', 'Virtus Pro');
INSERT INTO souperi_v VALUES ('2', 'Virtus Pro');
INSERT INTO souperi_v VALUES ('3', 'Virtus Pro');
INSERT INTO souperi_v VALUES ('1', 'Natus Vincere');
INSERT INTO souperi_v VALUES ('2', 'Natus Vincere');
INSERT INTO souperi_v VALUES ('3', 'Natus Vincere');
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 196 msec.
```

```sql
INSERT INTO trener VALUES ('87676768', '2008-7-16');
INSERT INTO trener VALUES ('44993300', '2009-12-3');
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 136 msec.
```

```sql
INSERT INTO hrac VALUES ('12345678', '4.2', '87676768');
INSERT INTO hrac VALUES ('54637825', '4.0', '87676768');
INSERT INTO hrac VALUES ('85736409', '4.7', '87676768');
INSERT INTO hrac VALUES ('74659023', '4.5', '87676768');
INSERT INTO hrac VALUES ('46395712', '4.1', '87676768');
INSERT INTO hrac VALUES ('55555555', '4.2', '44993300');
INSERT INTO hrac VALUES ('66774488', '4.4', '44993300');
INSERT INTO hrac VALUES ('88337755', '4.3', '44993300');
INSERT INTO hrac VALUES ('12345432', '4.6', '44993300');
INSERT INTO hrac VALUES ('87504738', '4.0', '44993300');
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 65 msec.
```

```sql
INSERT INTO je_spoluhracem VALUES ('12345678', '54637825');
INSERT INTO je_spoluhracem VALUES ('12345678', '85736409');
INSERT INTO je_spoluhracem VALUES ('12345678', '74659023');
INSERT INTO je_spoluhracem VALUES ('12345678', '46395712');
INSERT INTO je_spoluhracem VALUES ('54637825', '12345678');
INSERT INTO je_spoluhracem VALUES ('54637825', '85736409');
INSERT INTO je_spoluhracem VALUES ('54637825', '74659023');
INSERT INTO je_spoluhracem VALUES ('54637825', '46395712');
INSERT INTO je_spoluhracem VALUES ('85736409', '12345678');
INSERT INTO je_spoluhracem VALUES ('85736409', '54637825');
INSERT INTO je_spoluhracem VALUES ('85736409', '74659023');
INSERT INTO je_spoluhracem VALUES ('85736409', '46395712');
INSERT INTO je_spoluhracem VALUES ('74659023', '12345678');
INSERT INTO je_spoluhracem VALUES ('74659023', '54637825');
INSERT INTO je_spoluhracem VALUES ('74659023', '85736409');
INSERT INTO je_spoluhracem VALUES ('74659023', '46395712');
INSERT INTO je_spoluhracem VALUES ('46395712', '12345678');
INSERT INTO je_spoluhracem VALUES ('46395712', '54637825');
INSERT INTO je_spoluhracem VALUES ('46395712', '85736409');
INSERT INTO je_spoluhracem VALUES ('46395712', '74659023');
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 107 msec.
```

```
INSERT INTO specializace VALUES
('87676768', 'Utocna taktika v pripade vedeni, jinak obranna taktika.');
INSERT INTO specializace VALUES
('44993300', 'Obranna taktika v pripade vedeni, jinak utocna taktika.');
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 91 msec.
```

```
INSERT INTO strategie VALUES ('12345678', 'sniper, obrana');
INSERT INTO strategie VALUES ('54637825', 'smg, utok');
INSERT INTO strategie VALUES ('85736409', 'utocna puska, obrana i utok');
INSERT INTO strategie VALUES ('74659023', 'utocna puska, utok');
INSERT INTO strategie VALUES ('46395712', 'utocna puska, obrana');
INSERT INTO strategie VALUES ('55555555', 'sniper, obrana');
INSERT INTO strategie VALUES ('66774488', 'utocna puska, obrana');
INSERT INTO strategie VALUES ('88337755', 'brokovnice, obrana');
INSERT INTO strategie VALUES ('12345432', 'utocna puska, obrana i utok');
INSERT INTO strategie VALUES ('87504738', 'utocna puska, utok');
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 112 msec.
```

```
INSERT INTO kolo (cislo, id_zapasu, vitezny_tym, nejlepsi_hrac, delka)
VALUES ('1', '1', 'Virtus Pro', 'kawasaki', '2');
INSERT INTO kolo (cislo, id_zapasu, vitezny_tym, nejlepsi_hrac, delka)
VALUES ('2', '1', 'Virtus Pro', 'nowa', '3');
INSERT INTO kolo (cislo, id_zapasu, vitezny_tym, nejlepsi_hrac, delka)
VALUES ('3', '1', 'Virtus Pro', 'zeli', '3');
INSERT INTO kolo (cislo, id_zapasu, vitezny_tym, nejlepsi_hrac, delka)
VALUES ('4', '1', 'Virtus Pro', 'wojtek', '3');
INSERT INTO kolo (cislo, id_zapasu, vitezny_tym, nejlepsi_hrac, delka)
VALUES ('5', '1', 'Virtus Pro', 'dabrowski', '3');
INSERT INTO kolo (cislo, id_zapasu, vitezny_tym, nejlepsi_hrac, delka)
VALUES ('6', '1', 'Virtus Pro', 'nowa', '3');
INSERT INTO kolo (cislo, id_zapasu, vitezny_tym, nejlepsi_hrac, delka)
VALUES ('7', '1', 'Virtus Pro', 'kawasaki', '2');
INSERT INTO kolo (cislo, id_zapasu, vitezny_tym, nejlepsi_hrac, delka)
VALUES ('8', '1', 'Virtus Pro', 'zeli', '3');
INSERT INTO kolo (cislo, id_zapasu, vitezny_tym, nejlepsi_hrac, delka)
VALUES ('9', '1', 'Virtus Pro', 'kawasaki', '3');
INSERT INTO kolo (cislo, id_zapasu, vitezny_tym, nejlepsi_hrac, delka)
VALUES ('10', '1', 'Virtus Pro', 'dabrowski', '3');
INSERT INTO kolo (cislo, id_zapasu, vitezny_tym, nejlepsi_hrac, delka)
VALUES ('11', '1', 'Virtus Pro', 'nowa', '3');
INSERT INTO kolo (cislo, id_zapasu, vitezny_tym, nejlepsi_hrac, delka)
VALUES ('12', '1', 'Virtus Pro', 'kawasaki', '3');
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 91 msec.
```

```
INSERT INTO hraje VALUES ('12345678', '1');
INSERT INTO hraje VALUES ('54637825', '1');
INSERT INTO hraje VALUES ('85736409', '1');
INSERT INTO hraje VALUES ('74659023', '1');
INSERT INTO hraje VALUES ('46395712', '1');
INSERT INTO hraje VALUES ('55555555', '1');
INSERT INTO hraje VALUES ('66774488', '1');
INSERT INTO hraje VALUES ('88337755', '1');
INSERT INTO hraje VALUES ('12345432', '1');
INSERT INTO hraje VALUES ('87504738', '1');

INSERT INTO hraje VALUES ('12345678', '2');
INSERT INTO hraje VALUES ('54637825', '2');
INSERT INTO hraje VALUES ('85736409', '2');
INSERT INTO hraje VALUES ('74659023', '2');
INSERT INTO hraje VALUES ('46395712', '2');
INSERT INTO hraje VALUES ('55555555', '2');
INSERT INTO hraje VALUES ('66774488', '2');
INSERT INTO hraje VALUES ('88337755', '2');
INSERT INTO hraje VALUES ('12345432', '2');
INSERT INTO hraje VALUES ('87504738', '2');

INSERT INTO hraje VALUES ('12345678', '3');
INSERT INTO hraje VALUES ('54637825', '3');
```

Data Output Messages Notifications

```
INSERT 0 1

Query returned successfully in 65 msec.
```

Pro tabulky Zapas a Kolo jsem se rozhodl, že bude lepší přejmenovat atribut delka na delka_min, protože délka kola či zápasu je v minutách.

ALTER TABLE zapas RENAME COLUMN delka TO delka_min;

ALTER TABLE kolo RENAME COLUMN delka TO delka_min;

Data Output Messages Notifications

```
ALTER TABLE

Query returned successfully in 75 msec.
```

Níže přikládám dotazy na data z databáze s jejich vysvětlením a obrázky výsledků dotazů.

1. Toto je dotaz na vnější spojení tabulek v tabulce Profesional. Používá LEFT OUTER JOIN. Pro hráče jsou ve sloupci zkusenosti_od null hodnoty, pro trenéry jsou zde hodnoty uložené v databázi.

```
SELECT cislo_smlouvy, jmeno, prijmeni, zkusenosti_od FROM profesional
    LEFT OUTER JOIN trener USING (cislo_smlouvy);
```

Data Output | Messages | Notifications

	cislo_smlouvy integer	jmeno character varying (50)	prijmeni character varying (50)	zkusenosti_od date
1	87676768	Grzegorz	Brzeczyszczykiewicz	2008-07-16
2	44993300	Sergey	Baracuda	2009-12-03
3	12345678	Jan	Kowalski	[null]
4	46395712	Tomasz	Dąbrowski	[null]
5	85736409	Mateusz	Zieliński	[null]
6	12345432	Maxim	Sokolov	[null]
7	55555555	Dmitry	Ivanov	[null]
8	88337755	Alexander	Kuznetsov	[null]
9	87504738	Sergey	Popov	[null]
10	54637825	Wojciech	Nowak	[null]
11	66774488	Ivan	Petrov	[null]
12	74659023	Jakub	Wójcik	[null]

2. Toto je dotaz na vnitřní spojení tabulek se stejnými parametry jako v dotazu předtím. Používá INNER JOIN. Výsledek se liší tím, že zde už nejsou data hráčů, kteří nemají uložená data ve sloupci zkusenosti_od. Tabulka Hrac tedy ani nemá vůbec zkusenosti_od jako atribut.

```
SELECT cislo_smlouvy, jmeno, prijmeni, zkusenosti_od FROM profesional
    INNER JOIN trener USING (cislo_smlouvy);
```

Data Output | Messages | Notifications

	cislo_smlouvy integer	jmeno character varying (50)	prijmeni character varying (50)	zkusenosti_od date
1	87676768	Grzegorz	Brzeczyszczykiewicz	2008-07-16
2	44993300	Sergey	Baracuda	2009-12-03

3. Dotaz pokrývá podmínku na data pomocí WHERE. Výsledkem jsou hráči, kteří jsou pod zmíněným trenérem u WHERE klauzule.

```
SELECT * FROM hrac
        WHERE trener = '87676768';
```

Data Output | Messages | Notifications

	cislo_smlouvy [PK] integer	hodnoceni real	trener integer
1	12345678	4.2	87676768
2	54637825	4	87676768
3	85736409	4.7	87676768
4	74659023	4.5	87676768
5	46395712	4.1	87676768

4. Tento dotaz je směřován na agregaci dat s podmínkou. Vybíráni jsou jen hráči (jejich přezdívky) z tabulky Kolo, kteří byli nejlepšími hráči v nějakých kolech aspoň 5krát.

```
SELECT nejlepsi_hrac, COUNT(*) mvp FROM kolo
        GROUP BY nejlepsi_hrac
        HAVING COUNT(*) > 4;
```

Data Output | Messages | Notifications

	nejlepsi_hrac character varying (50)	mvp bigint
1	zeli	8
2	wojtek	5
3	nowa	6
4	dabrowski	5
5	kawasaki	8

5. Dotaz pokrývá řazení a stránkování. Z tabulky Profesional jsou vybírány jména hráčů nejdříve seřazena vzestupně podle příjmení a jsou z nich vybírány 4 řádky od 5. řádku už v seřazené tabulce.

```
SELECT jmeno, prijmeni FROM profesional

    ORDER BY prijmeni ASC

    LIMIT 4 OFFSET 4;
```

Data Output | Messages | Notifications

	jmeno character varying (50)	prijmeni character varying (50)
1	Jan	Kowalski
2	Alexander	Kuznetsov
3	Wojciech	Nowak
4	Ivan	Petrov

6. Zde je pokryta množinová operace, konkrétně UNION. Tento dotaz spojí jména týmů z tabulky Tym, které se účastnily prvního turnaje, opět s týmy z tabulky Tym, které se účastnily druhého turnaje, a odstraní duplicitní řádky.

```
SELECT nazev FROM tym NATURAL JOIN se_ucastni

WHERE se_ucastni.nazev_turnaje = 'ESL Majors Series One'

UNION

SELECT nazev FROM tym NATURAL JOIN se_ucastni

WHERE se_ucastni.nazev_turnaje = 'ESL Majors Series Two';
```

Data Output | Messages | Notifications

	nazev character varying (50)
1	Astralis
2	FaZe Clan
3	Fnatic
4	Natus Vincere
5	Virtus Pro

7. Tento dotaz je na použití vnořeného SELECTU. Vybírá čísla kol a nejlepší hráče (jejich přezdívky) jen z kol, které mají cizí klíč id_zapasu roven 1. To vlastně znamená výběr nejlepších hráčů všech kolech prvního zápasu turnaje.

```
SELECT cislo, nejlepsi_hrac FROM kolo
    WHERE id_zapasu IN (SELECT id_zapasu FROM zapas
        WHERE id_zapasu = 1);
```

Data Output | Messages | Notifications

	cislo integer	nejlepsi_hrac character varying (50)
1	1	kawasaki
2	2	nowa
3	3	zeli
4	4	wojtek
5	5	dabrowski
6	6	nowa
7	7	kawasaki
8	8	zeli
9	9	kawasaki
10	10	dabrowski
11	11	nowa
12	12	kawasaki
13	13	wojtek
14	14	zeli
15	15	zeli
16	16	kawasaki

Kvůli vložení přibližně 32 tisíc provozních dat do jedné tabulky jsem musel vložit i další relevantní data do ostatních tabulek. Do tabulek Turnaj, Tym, Organizator, Cena, Se_ucastni, Trener a Specializace jsem vkládal přímo pomocí SQL. Zde už jen obrázky dotazů v pořadí zmíněných tabulek přikládám níže.

```
1 INSERT INTO turnaj VALUES ('ESL Majors Series Two', 'Nemecko', 'Berlin', '2015-7-10');
2 INSERT INTO turnaj VALUES ('ESL Majors Series Three', 'Francie', 'Pariz', '2016-7-19');
3 INSERT INTO turnaj VALUES ('ESL Majors Series Four', 'Rakousko', 'Linz', '2017-7-27');
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 177 msec.
```

```
1 INSERT INTO tym VALUES ('Astralis', 'Dansko', '2012-7-9', '4.8');
2 INSERT INTO tym VALUES ('FaZe Clan', 'Spojene Staty Americke', '2012-9-15', '4.7');
3 INSERT INTO tym VALUES ('Fnatic', 'Velka Britanie', '2012-11-21', '4.4');
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 142 msec.
```

```
INSERT INTO organizator VALUES ('ESL Majors Series Two', 'Hans', 'Zimmer');
INSERT INTO organizator VALUES ('ESL Majors Series Two', 'Frederick', 'Mann');
INSERT INTO organizator VALUES ('ESL Majors Series Three', 'Charles', 'Bernard');
INSERT INTO organizator VALUES ('ESL Majors Series Three', 'Antoine', 'Petit');
INSERT INTO organizator VALUES ('ESL Majors Series Three', 'Olivier', 'Dubois');
INSERT INTO organizator VALUES ('ESL Majors Series Four', 'Hans', 'Landa');
INSERT INTO organizator VALUES ('ESL Majors Series Four', 'Tomas', 'Muller');
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 169 msec.
```

```
INSERT INTO cena VALUES ('Berlin Masters 2015', '2015-8-25', 'zlaty pohar', 'ESL Majors Series Two', 'Astralis');
INSERT INTO cena VALUES ('Paris Masters 2016', '2016-8-30', 'zlaty pohar', 'ESL Majors Series Three', 'Natus Vincere');
INSERT INTO cena VALUES ('Linz Masters 2017', '2017-9-9', 'zlaty pohar', 'ESL Majors Series Four', 'FaZe Clan');
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 225 msec.
```

```
INSERT INTO se_ucastni VALUES ('ESL Majors Series One', 'Astralis');
INSERT INTO se_ucastni VALUES ('ESL Majors Series One', 'FaZe Clan');
INSERT INTO se_ucastni VALUES ('ESL Majors Series One', 'Fnatic');
INSERT INTO se_ucastni VALUES ('ESL Majors Series Two', 'Virtus Pro');
INSERT INTO se_ucastni VALUES ('ESL Majors Series Two', 'Natus Vincere');
INSERT INTO se_ucastni VALUES ('ESL Majors Series Two', 'Astralis');
INSERT INTO se_ucastni VALUES ('ESL Majors Series Two', 'FaZe Clan');
INSERT INTO se_ucastni VALUES ('ESL Majors Series Two', 'Fnatic');
INSERT INTO se_ucastni VALUES ('ESL Majors Series Three', 'Virtus Pro');
INSERT INTO se_ucastni VALUES ('ESL Majors Series Three', 'Natus Vincere');
INSERT INTO se_ucastni VALUES ('ESL Majors Series Three', 'Astralis');
INSERT INTO se_ucastni VALUES ('ESL Majors Series Three', 'FaZe Clan');
INSERT INTO se_ucastni VALUES ('ESL Majors Series Three', 'Fnatic');
INSERT INTO se_ucastni VALUES ('ESL Majors Series Four', 'Virtus Pro');
INSERT INTO se_ucastni VALUES ('ESL Majors Series Four', 'Natus Vincere');
INSERT INTO se_ucastni VALUES ('ESL Majors Series Four', 'Astralis');
INSERT INTO se_ucastni VALUES ('ESL Majors Series Four', 'FaZe Clan');
INSERT INTO se_ucastni VALUES ('ESL Majors Series Four', 'Fnatic');
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 1 secs 340 msec.
```

```
INSERT INTO trener VALUES ('34152002', '2012-1-2');
INSERT INTO trener VALUES ('55698593', '2012-12-5');
INSERT INTO trener VALUES ('59438139', '2012-6-25');
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 167 msec.
```

```
INSERT INTO specializace VALUES ('34152002', 'Utocna taktika v pripade vedeni, jinak obranna taktika.');
INSERT INTO specializace VALUES ('55698593', 'Obranna taktika v pripade vedeni, jinak utocna taktika.');
INSERT INTO specializace VALUES ('59438139', 'Utocna taktika v pripade vedeni, jinak obranna taktika.');
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 100 msec.
```

Pro zbylé tabulky (Profesional, Hrac, Strategie, Je_spoluhracem, Zapas, Kolo, Souperi_v, Hraje) jsem použil Pythonovských skriptů, které jako obrázky přikládám níže postupně v pořadí zmíněných tabulek. Udělal jsem tak, aby data zůstala relevantní pro zbytek databáze. Provozní data jsou využita v tabulce Hraje, je jich přibližně 36 tisíc.

```python
import random
import datetime
import psycopg2

1 usage
def generate_id():
    return random.randint( a: 10000000,  b: 99999999)

1 usage
def choose_country(countries, i):
    if i < 6: return countries[0]
    elif i < 12: return countries[1]
    else: return countries[2]

1 usage
def generate_player_data(i):
    first_names = ["Lars", "Mikkel", "Mads", "Peter", "Rasmus", "Magnus",
                   "Oliver", "Harry", "Jack", "James", "Charlie", "George",
                   "Noah", "Liam", "William", "James", "Lucas", "Benjamin"]
    last_names = ["Larsen", "Sorensen", "Jensen", "Nielsen", "Rasmussen", "Lagerstedt",
                  "Smith", "Jones", "Brown", "Williams", "Taylor", "Jackson",
                  "Johnson", "Miller", "Smith", "Rodriguez", "Davis", "Holland"]
    nicknames = ["noob", "hax", "camper", "smurf", "tryhard", "afk",
                 "tank", "healer", "dps", "fragger", "jungler", "carry",
                 "cheeseball", "scrub", "sweat", "troll", "bot", "pwned"]
    countries = ["Dánsko", "Spojené Státy Americké", "Velká Británie"]
    date = datetime.date(random.randint( a: 1985,  b: 1999), random.randint( a: 1,  b: 12), random.randint( a: 1,  b: 28))
    format_date = date.strftime("%Y-%m-%d")
    since_date = datetime.date(random.randint( a: 2013,  b: 2013), random.randint( a: 1,  b: 12), random.randint( a: 1,  b: 28))
    format_since_date = since_date.strftime("%Y-%m-%d")
    return {
        "cislo_smlouvy": generate_id(),
        "jmeno": first_names[i],
        "prijmeni": last_names[i],
        "prezdivka_ve_hre": nicknames[i],
        "datum_narozeni": format_date,
        "zeme_puvodu": choose_country(countries, i),
        "clenem_od": format_since_date
    }

conn = psycopg2.connect(
    database="cafoulu1",
    user="cafoulu1",
    password="<PASSWORD>",
    host="slon.felk.cvut.cz",
    port="5432"
)
cursor = conn.cursor()
teams = ["Astralis", "FaZe Clan", "Fnatic"]
for i in range(18):
    player_data = generate_player_data(i)
    if i < 6: player_data['nazev_tymu'] = teams[0]
    elif i < 12: player_data['nazev_tymu'] = teams[1]
    else: player_data['nazev_tymu'] = teams[2]
    cursor.execute("""
        INSERT INTO profesional (cislo_smlouvy, jmeno, prijmeni, prezdivka_ve_hre, datum_narozeni, zeme_puvodu, clenem_od, nazev_tymu)
        VALUES (%s, %s, %s, %s, %s, %s, %s, %s)""", tuple(player_data.values()))
conn.commit()
conn.close()
```

```python
import random
import psycopg2

3 usages
def generate_player_data(data, trener):
    return {
        "cislo_smlouvy": data[0],
        "hodnoceni": random.uniform( a: 4.0,  b: 5.0),
        "trener": trener[0]
    }

conn = psycopg2.connect(
    database="cafoulu1",
    user="cafoulu1",
    password="<PASSWORD>",
    host="slon.felk.cvut.cz",
    port="5432"
)
cursor = conn.cursor()
cursor.execute("SELECT cislo_smlouvy FROM profesional OFFSET 12")
results = cursor.fetchall()
data = []
for row in results:
    data.append(row)
for i in range(18):
    if i < 5: player_data = generate_player_data(data[i], data[5])
    elif 5 < i < 11: player_data = generate_player_data(data[i], data[11])
    elif 11 < i < 17: player_data = generate_player_data(data[i], data[17])
    else: continue
    cursor.execute("""
        INSERT INTO hrac (cislo_smlouvy, hodnoceni, trener)
        VALUES (%s, %s, %s)""", tuple(player_data.values()))
conn.commit()
conn.close()
```

```python
import random
import psycopg2

1 usage
def generate_player_data(data):
    strategy = ["sniper", "smg", "brokovnice", "utocna puska"]
    strategy2 = ["obrana", "utok", "obrana i utok"]
    return {
        "hrac": data[0],
        "strategie": random.choice(strategy) + ", " + random.choice(strategy2)
    }

conn = psycopg2.connect(
    database="cafoulu1",
    user="cafoulu1",
    password="<PASSWORD>",
    host="slon.felk.cvut.cz",
    port="5432"
)
cursor = conn.cursor()
cursor.execute("SELECT cislo_smlouvy FROM hrac OFFSET 10")
results = cursor.fetchall()
data = []
for row in results:
    data.append(row)
for i in range(15):
    player_data = generate_player_data(data[i])
    cursor.execute("""
        INSERT INTO strategie (hrac, strategie)
        VALUES (%s, %s)""", tuple(player_data.values()))
conn.commit()
conn.close()
```

```python
import psycopg2

conn = psycopg2.connect(
    database="cafoulu1",
    user="cafoulu1",
    password="<PASSWORD>",
    host="slon.felk.cvut.cz",
    port="5432"
)
cursor = conn.cursor()
cursor.execute("SELECT cislo_smlouvy FROM hrac OFFSET 10")
results = cursor.fetchall()
players = [row[0] for row in results]
teams = [players[i:i+5] for i in range(0, len(players), 5)]
for team in teams:
    for player1 in team:
        for player2 in team:
            if player1 != player2:
                cursor.execute("""
                    INSERT INTO je_spoluhracem (hrac, spoluhrac)
                    VALUES (%s, %s)""", (player1, player2))
conn.commit()
conn.close()
```

```python
import random
import itertools
import psycopg2
import datetime

conn = psycopg2.connect(
    database="cafoulu1",
    user="cafoulu1",
    password="<PASSWORD>",
    host="slon.felk.cvut.cz",
    port="5432"
)
cursor = conn.cursor()
cursor.execute("SELECT nazev FROM turnaj")
results = cursor.fetchall()
turnaje = [row[0] for row in results]
teams = ["Virtus Pro", "Natus Vincere", "Astralis", "FaZe Clan", "Fnatic"]
maps = ["Ancient", "Anubis", "Inferno", "Mirage", "Nuke", "Overpass", "Vertigo"]
1 usage
def generate_matches():
    for turnaj in turnaje:
        team_combinations = list(itertools.combinations(teams, r: 2))
        for _ in range(3):
            for (tym_A, tym_B) in team_combinations:
                mapa = random.choice(maps)
                delka_min = random.randint( a: 40, b: 60)
                datum_konani = random_date_for_tournament(turnaj)
                match_data = {
                    "mapa": mapa,
                    "tym_A": tym_A,
                    "tym_B": tym_B,
                    "datum_konani": datum_konani,
                    "vitezny_tym": tym_A,
                    "delka_min": delka_min,
                    "nazev_turnaje": turnaj
                }
                cursor.execute("""
                    INSERT INTO zapas (mapa, tym_a, tym_b, datum_konani, vitezny_tym, delka_min, nazev_turnaje)
                    VALUES (%s, %s, %s, %s, %s, %s, %s)""", tuple(match_data.values()))
                conn.commit()

1 usage
def random_date_for_tournament(turnaj):
    global year
    if turnaj.endswith("One"):
        year = 2014
    elif turnaj.endswith("Two"):
        year = 2015
    elif turnaj.endswith("Three"):
        year = 2016
    elif turnaj.endswith("Four"):
        year = 2017
    datum_konani = datetime.date(year, random.randint( a: 7, b: 8), random.randint( a: 1, b: 28))
    return datum_konani.strftime("%Y-%m-%d")

generate_matches()
conn.close()
```

```python
import random
import psycopg2

conn = psycopg2.connect(
    database="cafoulu1",
    user="cafoulu1",
    password="<PASSWORD>",
    host="slon.felk.cvut.cz",
    port="5432"
)
cursor = conn.cursor()
cursor.execute("SELECT id, tym_a, tym_b FROM zapas OFFSET 3")
results = cursor.fetchall()
zapasy = [row[0] for row in results]
tymy = [(row[1], row[2]) for row in results]
nicknames1 = ["kawasaki", "nowa", "zeli", "wojtek", "dabrowski"]
nicknames2 = ["hey_ivan", "petra", "kuznyak", "sokol", "popovski"]
nicknames3 = ["noob", "hax", "camper", "smurf", "tryhard"]
nicknames4 = ["tank", "healer", "dps", "fragger", "jungler"]
nicknames5 = ["cheeseball", "scrub", "sweat", "troll", "bot"]
id_zapasu = 0
nejlepsi_hrac = ""
for zapas in zapasy:
    for i in range(1, 31):
        cislo = i
        delka_min = random.randint( a: 1,  b: 3)
        if i < 15:
            vitezny_tym = tymy[id_zapasu][1]
            if tymy[id_zapasu][1] == "Virtus Pro": nejlepsi_hrac = random.choice(nicknames1)
            elif tymy[id_zapasu][1] == "Natus Vincere": nejlepsi_hrac = random.choice(nicknames2)
            elif tymy[id_zapasu][1] == "Astralis": nejlepsi_hrac = random.choice(nicknames3)
            elif tymy[id_zapasu][1] == "FaZe Clan": nejlepsi_hrac = random.choice(nicknames4)
            elif tymy[id_zapasu][1] == "Fnatic": nejlepsi_hrac = random.choice(nicknames5)
        else:
            vitezny_tym = tymy[id_zapasu][0]
            if tymy[id_zapasu][0] == "Virtus Pro": nejlepsi_hrac = random.choice(nicknames1)
            elif tymy[id_zapasu][0] == "Natus Vincere": nejlepsi_hrac = random.choice(nicknames2)
            elif tymy[id_zapasu][0] == "Astralis": nejlepsi_hrac = random.choice(nicknames3)
            elif tymy[id_zapasu][0] == "FaZe Clan": nejlepsi_hrac = random.choice(nicknames4)
            elif tymy[id_zapasu][0] == "Fnatic": nejlepsi_hrac = random.choice(nicknames5)
        kolo_data = {
            "cislo": cislo,
            "id_zapasu": id_zapasu + 4,
            "vitezny_tym": vitezny_tym,
            "nejlepsi_hrac": nejlepsi_hrac,
            "delka_min": delka_min
        }
        cursor.execute("""
        INSERT INTO kolo (cislo, id_zapasu, vitezny_tym, nejlepsi_hrac, delka_min)
        VALUES (%s, %s, %s, %s, %s)""", tuple(kolo_data.values()))
        conn.commit()
    id_zapasu += 1
conn.close()
```

```python
import psycopg2

conn = psycopg2.connect(
    database="cafoulu1",
    user="cafoulu1",
    password="<PASSWORD>",
    host="slon.felk.cvut.cz",
    port="5432"
)
cursor = conn.cursor()
cursor.execute("SELECT id, tym_A, tym_b FROM zapas ORDER BY id ASC")
results = cursor.fetchall()
for result in results:
    cursor.execute("""INSERT INTO souperi_v VALUES (%s, %s)""", (result[0], result[1]))
    cursor.execute("""INSERT INTO souperi_v VALUES (%s, %s)""", (result[0], result[2]))
    conn.commit()
conn.close()
```

```python
import psycopg2

conn = psycopg2.connect(
    database="cafoulu1",
    user="cafoulu1",
    password="<PASSWORD>",
    host="slon.felk.cvut.cz",
    port="5432"
)
cursor = conn.cursor()
cursor.execute("SELECT id, id_zapasu FROM kolo ORDER BY id ASC OFFSET 48")
result1 = cursor.fetchall()
cursor.execute("SELECT id_zapasu, nazev_tymu FROM souperi_v ORDER BY id_zapasu ASC OFFSET 6")
result2 = cursor.fetchall()
cursor.execute("SELECT cislo_smlouvy FROM hrac")
result3 = cursor.fetchall()
kola = [(row[0], row[1]) for row in result1]
zapasy = [(row[0], row[1]) for row in result2]
hraci = [row[0] for row in result3]
query = "INSERT INTO hraje (hrac, id_kola) VALUES (%s, %s)"
data = []
for zapas in zapasy:
    for kolo in kola:
        if kolo[1] == zapas[0]:
            if zapas[1] == "Virtus Pro":
                for i in range(5):
                    data.append((hraci[i], kolo[0]))
            elif zapas[1] == "Natus Vincere":
                for i in range(5, 10):
                    data.append((hraci[i], kolo[0]))
            elif zapas[1] == "Astralis":
                for i in range(10, 15):
                    data.append((hraci[i], kolo[0]))
            elif zapas[1] == "FaZe Clan":
                for i in range(15, 20):
                    data.append((hraci[i], kolo[0]))
            elif zapas[1] == "Fnatic":
                for i in range(20, 25):
                    data.append((hraci[i], kolo[0]))
cursor.executemany(query, data)
conn.commit()
conn.close()
```