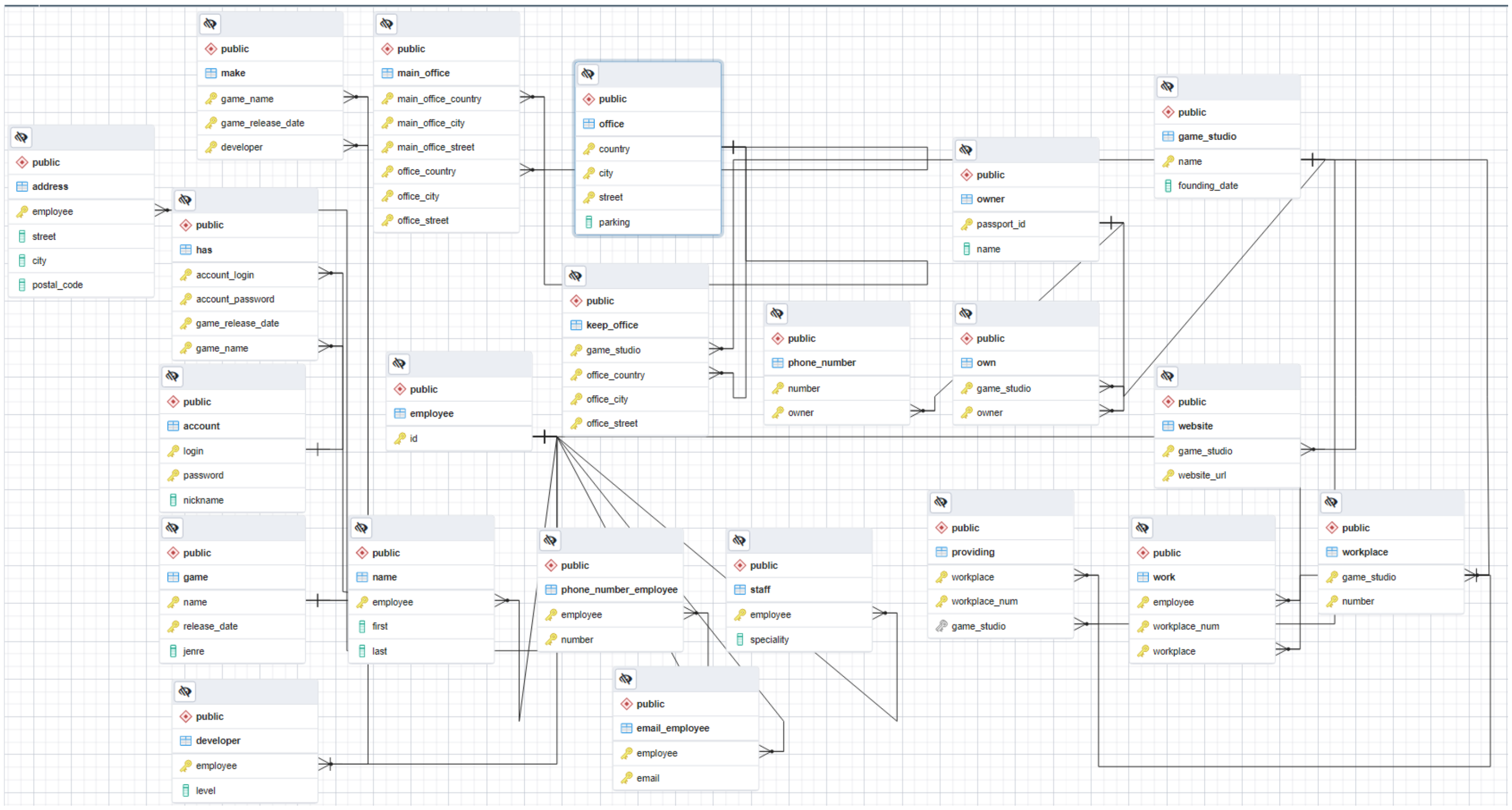
CP-3 SQL - Vytvoření databáze, dotazy na data



```
DROP TABLE IF EXIST owner;
DROP TABLE IF EXIST phone_number;
DROP TABLE IF EXIST game_studio;
DROP TABLE IF EXIST website;
DROP TABLE IF EXIST own;
DROP TABLE IF EXIST office;
DROP TABLE IF EXIST main_office;
DROP TABLE IF EXIST keep_office;
DROP TABLE IF EXIST workplace;
DROP TABLE IF EXIST providing;
DROP TABLE IF EXIST employee;
DROP TABLE IF EXIST name;
DROP TABLE IF EXIST address;
DROP TABLE IF EXIST phone_number_employee;
DROP TABLE IF EXIST email_employee;
DROP TABLE IF EXIST staff;
DROP TABLE IF EXIST developer;
DROP TABLE IF EXIST work;
DROP TABLE IF EXIST game;
DROP TABLE IF EXIST make;
DROP TABLE IF EXIST account;
DROP TABLE IF EXIST has;
```

```
CREATE TABLE owner (
  passport_ID  INTEGER PRIMARY KEY,
  name         VARCHAR (50) NOT NULL UNIQUE
);


CREATE TABLE phone_number (
  number       VARCHAR(50),
  owner        INTEGER,

  PRIMARY KEY (number, owner),

  CONSTRAINT phone_number_fk_owner FOREIGN KEY (owner) REFERENCES owner (passport_ID) ON
UPDATE CASCADE ON DELETE CASCADE
);


CREATE TABLE game_studio (
  name                 VARCHAR (50) PRIMARY KEY,
  founding_date        DATE NOT NULL,

  CHECK (founding_date > '1971-01-01')
);


CREATE TABLE website (
  game_studio          VARCHAR (50),
  website_url          VARCHAR (200),

  PRIMARY KEY (game_studio, website_url),

  CONSTRAINT website_fk_game_studio FOREIGN KEY (game_studio) REFERENCES game_studio (name)
ON UPDATE CASCADE ON DELETE CASCADE
);


CREATE TABLE own (
  game_studio          VARCHAR (50),
  owner                INTEGER,

  PRIMARY KEY (game_studio, owner),

  CONSTRAINT own_fk_game_studio FOREIGN KEY (game_studio) REFERENCES game_studio (name) ON
UPDATE CASCADE ON DELETE CASCADE,
  CONSTRAINT own_fk_owner FOREIGN KEY (owner) REFERENCES owner (passport_ID) ON UPDATE
CASCADE ON DELETE CASCADE
);


CREATE TABLE office (
  country      VARCHAR (50),
  city         VARCHAR (50),
  street       VARCHAR (50),
  parking      BOOLEAN NOT NULL,

  PRIMARY KEY (country, city, street)
);
```

```
CREATE TABLE main_office (
  main_office_country           VARCHAR (50),
  main_office_city              VARCHAR (50),
  main_office_street            VARCHAR (50),

  office_country                VARCHAR (50),
  office_city                   VARCHAR (50),
  office_street                 VARCHAR (50),

 PRIMARY KEY(main_office_country, main_office_city, main_office_street, office_country, office_city,
office_street),

  CONSTRAINT main_office_fk_office_1 FOREIGN KEY (main_office_country, main_office_city,
main_office_street) REFERENCES office (country, city, street) ON UPDATE CASCADE ON DELETE
CASCADE,
  CONSTRAINT main_office_fk_office_2 FOREIGN KEY (office_country, office_city, office_street)
REFERENCES office (country, city, street) ON UPDATE CASCADE ON DELETE CASCADE
);


CREATE TABLE keep_office (
  game_studio          VARCHAR (50),
  office_country       VARCHAR (50),
  office_city          VARCHAR (50),
  office_street        VARCHAR (50),

 PRIMARY KEY (game_studio, office_country, office_city, office_street),

  CONSTRAINT keep_office_fk_game_studio FOREIGN KEY (game_studio) REFERENCES game_studio
(name) ON UPDATE CASCADE ON DELETE CASCADE,
  CONSTRAINT keep_office_fk_office FOREIGN KEY (office_country, office_city, office_street)
REFERENCES office (country, city, street) ON UPDATE CASCADE ON DELETE CASCADE
);


CREATE TABLE workplace (
  game_studio          VARCHAR (50),
  number               INTEGER,

  PRIMARY KEY(game_studio, number),

  CONSTRAINT workplace_fk_game_studio FOREIGN KEY (game_studio) REFERENCES game_studio
(name) ON UPDATE CASCADE ON DELETE CASCADE
);


CREATE TABLE providing (
  workplace            VARCHAR (50),
  workplace_num        INTEGER,
  game_studio          VARCHAR (50),

PRIMARY KEY( workplace_num, workplace),

  CONSTRAINT providing_fk_workplace FOREIGN KEY (workplace, workplace_num) REFERENCES
workplace (game_studio, number) ON UPDATE CASCADE ON DELETE CASCADE,
  CONSTRAINT providing_fk_game_studio FOREIGN KEY (game_studio) REFERENCES game_studio
(name) ON UPDATE CASCADE ON DELETE CASCADE
);
```

```
CREATE TABLE employee (
  id      INTEGER PRIMARY KEY
);


CREATE TABLE name (
  employee     INTEGER PRIMARY KEY,
  first        VARCHAR (50) NOT NULL,
  last         VARCHAR (50) NOT NULL,

  CONSTRAINT name_fk_employee FOREIGN KEY (employee) REFERENCES employee (id) ON UPDATE
CASCADE ON DELETE CASCADE
);


CREATE TABLE address (
  employee              INTEGER PRIMARY KEY,
  street                VARCHAR (50) NOT NULL,
  city                  VARCHAR (50) NOT NULL,
  postal_code           VARCHAR (50) NOT NULL,

  CONSTRAINT address_fk_employee FOREIGN KEY (employee) REFERENCES employee (id) ON
UPDATE CASCADE ON DELETE CASCADE
);


CREATE TABLE phone_number_employee(
  employee     INTEGER,
  number       VARCHAR (50),

  PRIMARY KEY (employee, number),

  CONSTRAINT phone_number_employee_fk_employee FOREIGN KEY (employee) REFERENCES
employee (id) ON UPDATE CASCADE ON DELETE CASCADE
);


CREATE TABLE email_employee (
  employee              INTEGER,
  email                 VARCHAR (50),

  PRIMARY KEY (employee, email),

  CONSTRAINT email_employee_fk_employee FOREIGN KEY (employee) REFERENCES employee (id) ON
UPDATE CASCADE ON DELETE CASCADE
  );


CREATE TABLE staff (
  employee              INTEGER PRIMARY KEY,
  speciality            VARCHAR (50) NOT NULL,

  CONSTRAINT staff_fk_employee FOREIGN KEY (employee) REFERENCES employee (id) ON UPDATE
CASCADE ON DELETE CASCADE
  );
```

```
CREATE TABLE developer (
  employee            INTEGER PRIMARY KEY,
  level               VARCHAR (50) NOT NULL,

  CONSTRAINT developer_fk_employee FOREIGN KEY (employee) REFERENCES employee (id) ON
UPDATE CASCADE ON DELETE CASCADE
  );



CREATE TABLE work (
  employee            INTEGER,
  workplace_num       INTEGER,
  workplace           VARCHAR (50) NOT NULL,

PRIMARY KEY(employee, workplace_num, workplace),

  CONSTRAINT work_fk_employee FOREIGN KEY (employee) REFERENCES employee (id) ON UPDATE
CASCADE ON DELETE CASCADE,
  CONSTRAINT work_fk_workplace FOREIGN KEY (workplace_num, workplace) REFERENCES workplace
(number, game_studio) ON UPDATE CASCADE ON DELETE CASCADE
  );


CREATE TABLE game (
  name                VARCHAR (50),
  release_date        DATE,
  jenre               VARCHAR (50) NOT NULL,

  PRIMARY KEY (name, release_date)
  );


CREATE TABLE make (
  game_name                VARCHAR (50),
  game_release_date        DATE,
  developer                INTEGER,

  PRIMARY KEY (game_name, game_release_date, developer),

  CONSTRAINT make_fk_developer FOREIGN KEY (developer) REFERENCES developer (employee) ON
UPDATE CASCADE ON DELETE CASCADE,
  CONSTRAINT make_fk_game FOREIGN KEY (game_name, game_release_date) REFERENCES game
(name, release_date) ON UPDATE CASCADE ON DELETE CASCADE
  );

CREATE TABLE account (
  login               VARCHAR (50),
  password            VARCHAR (50),
  nickname            VARCHAR (50) NOT NULL,

  PRIMARY KEY (login, password)
  );
```

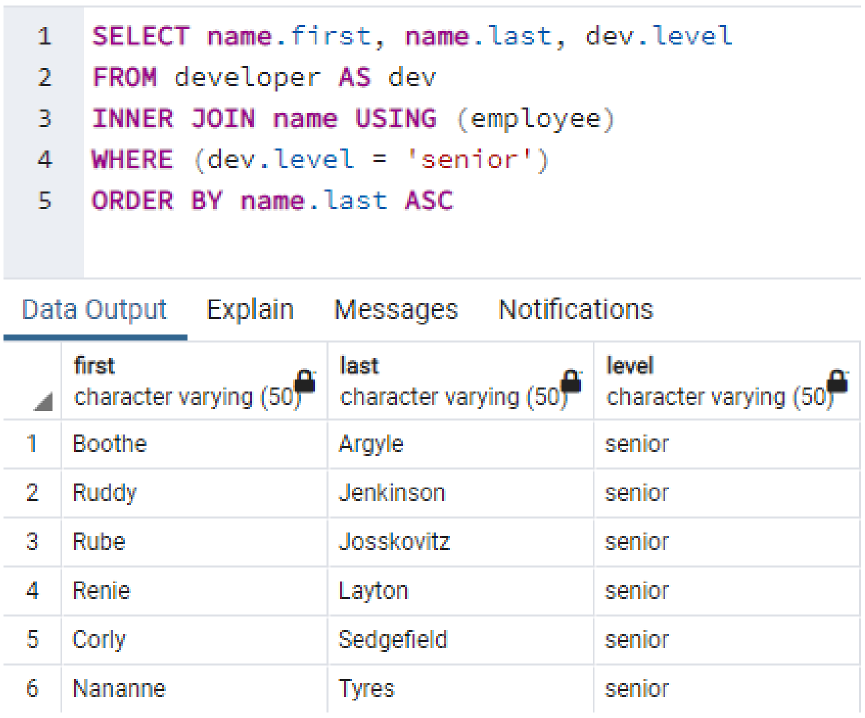
```
CREATE TABLE has (
  account_login                VARCHAR (50),
  account_password             VARCHAR (50),
  game_release_date            DATE,
  game_name                    VARCHAR (50),

  PRIMARY KEY (account_login, account_password, game_release_date, game_name),

  CONSTRAINT has_fk_account FOREIGN KEY (account_login, account_password) REFERENCES account
(login, password) ON UPDATE CASCADE ON DELETE CASCADE,
  CONSTRAINT has_fk_game FOREIGN KEY (game_name, game_release_date) REFERENCES game
(name, release_date) ON UPDATE CASCADE ON DELETE CASCADE
  );
```

/* get all señor developers and sort them alphabetically by last name */

```
SELECT name.first, name.last, dev.level
FROM developer AS dev
INNER JOIN name USING (employee)
WHERE (dev.level = 'senior')
ORDER BY name.last ASC
```



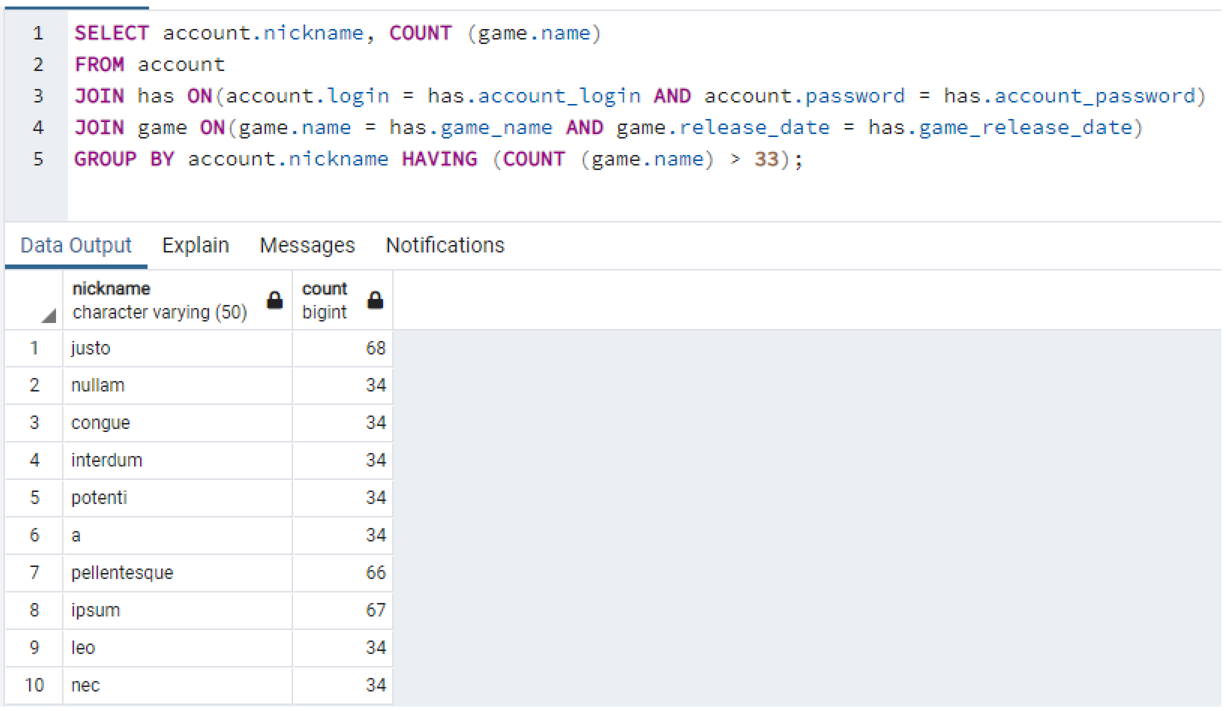
```
SELECT name.first, name.last, dev.level
FROM developer AS dev
INNER JOIN name USING (employee)
WHERE (dev.level = 'senior')
ORDER BY name.last ASC
```

Data Output | Explain | Messages | Notifications

	first character varying (50)	last character varying (50)	level character varying (50)
1	Boothe	Argyle	senior
2	Ruddy	Jenkinson	senior
3	Rube	Josskovitz	senior
4	Renie	Layton	senior
5	Corly	Sedgefield	senior
6	Nananne	Tyres	senior

/* get all nicknames of accounts with more than 33 games*/

```
SELECT account.nickname, COUNT (game.name)
FROM account
JOIN has ON(account.login = has.account_login AND account.password = has.account_password)
JOIN game ON(game.name = has.game_name AND game.release_date = has.game_release_date)
GROUP BY account.nickname HAVING (COUNT (game.name) > 33);
```



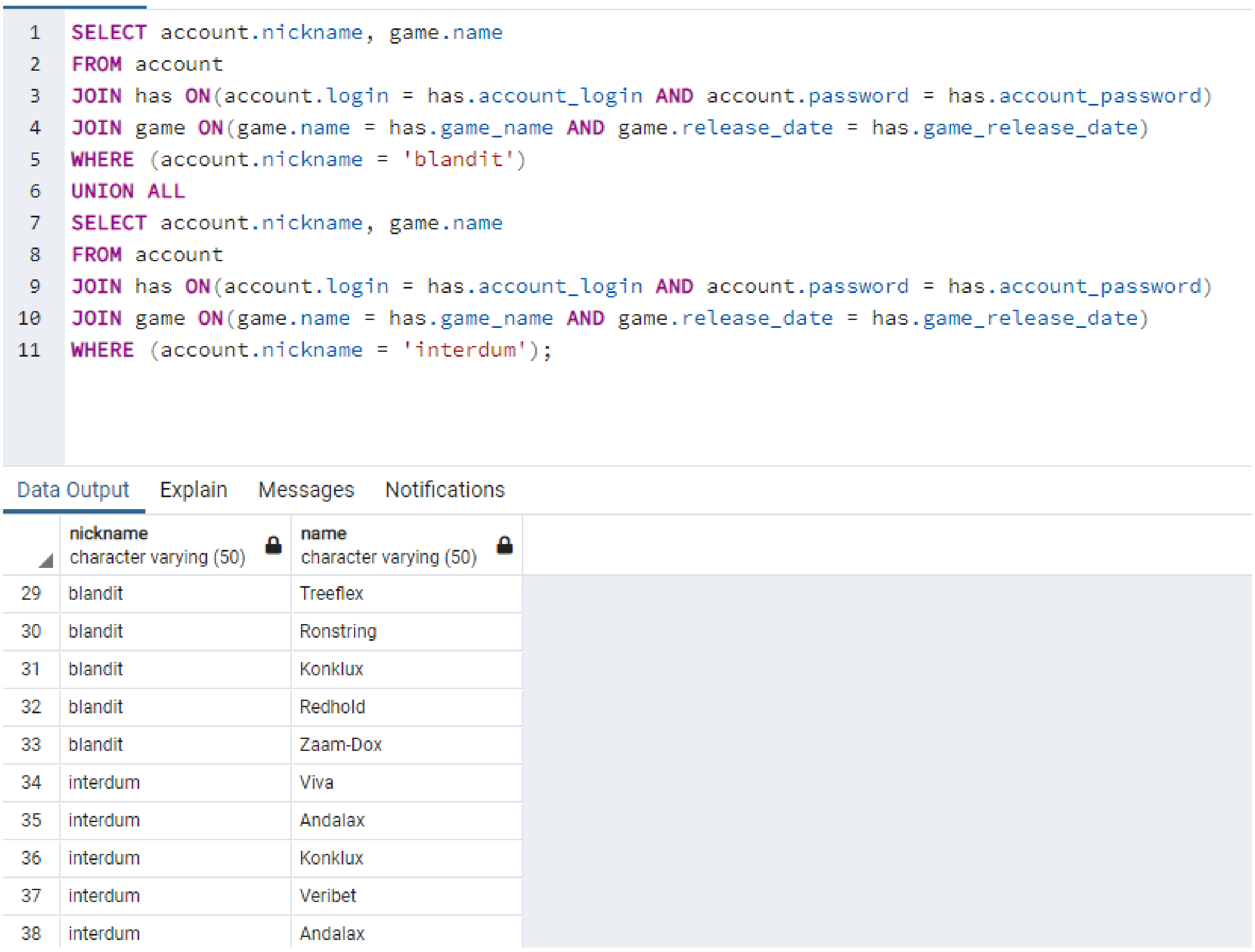
```
SELECT account.nickname, COUNT (game.name)
FROM account
JOIN has ON(account.login = has.account_login AND account.password = has.account_password)
JOIN game ON(game.name = has.game_name AND game.release_date = has.game_release_date)
GROUP BY account.nickname HAVING (COUNT (game.name) > 33);
```

Data Output | Explain | Messages | Notifications

	nickname character varying (50)	count bigint
1	justo	68
2	nullam	34
3	congue	34
4	interdum	34
5	potenti	34
6	a	34
7	pellentesque	66
8	ipsum	67
9	leo	34
10	nec	34

```sql
/* get games of two accounts */

SELECT account.nickname, game.name
FROM account
JOIN has ON(account.login = has.account_login AND account.password = has.account_password)
JOIN game ON(game.name = has.game_name AND game.release_date = has.game_release_date)
WHERE (account.nickname = 'blandit')
UNION ALL
SELECT account.nickname, game.name
FROM account
JOIN has ON(account.login = has.account_login AND account.password = has.account_password)
JOIN game ON(game.name = has.game_name AND game.release_date = has.game_release_date)
WHERE (account.nickname = 'interdum');
```

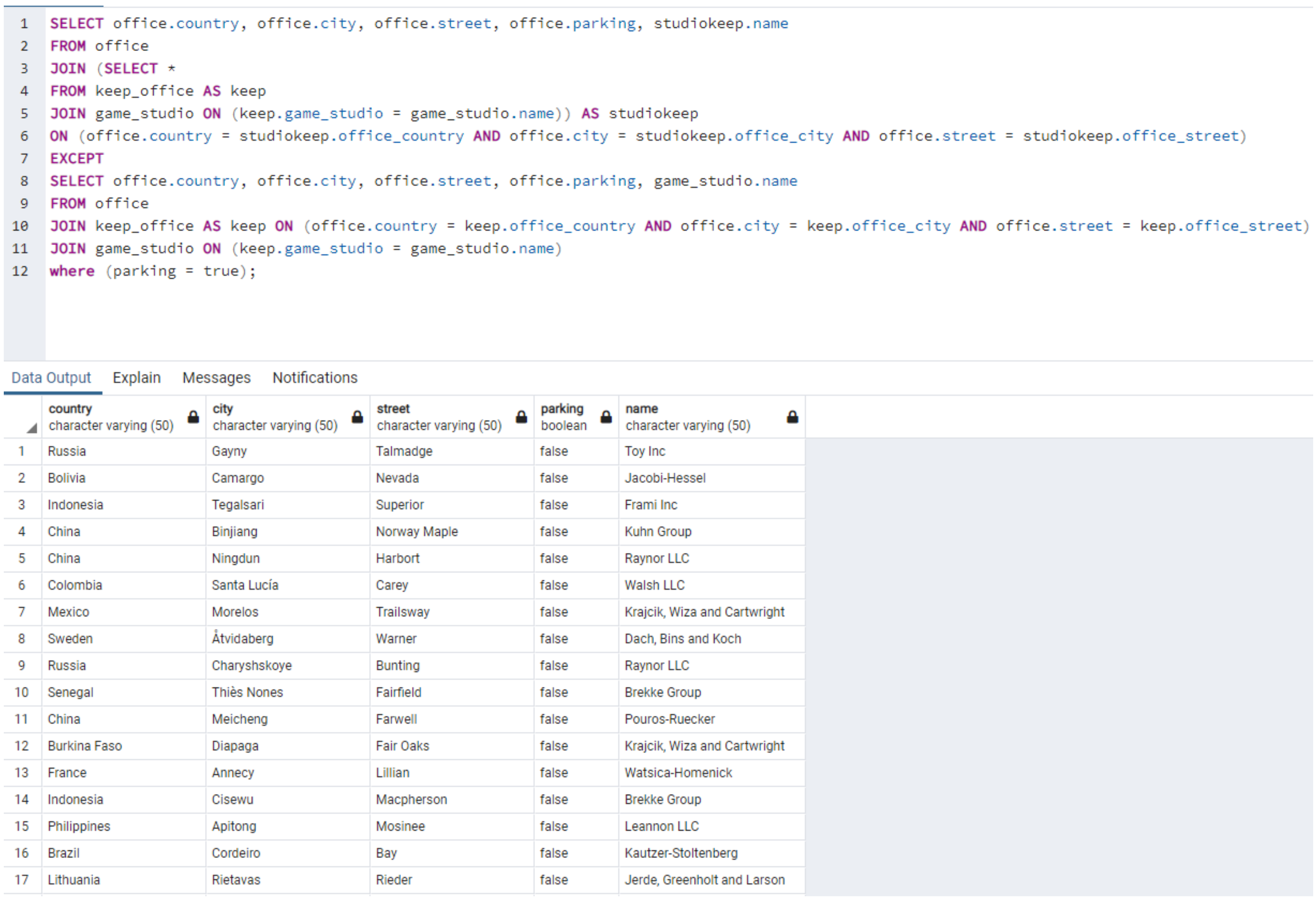


```sql
SELECT account.nickname, game.name
FROM account
JOIN has ON(account.login = has.account_login AND account.password = has.account_password)
JOIN game ON(game.name = has.game_name AND game.release_date = has.game_release_date)
WHERE (account.nickname = 'blandit')
UNION ALL
SELECT account.nickname, game.name
FROM account
JOIN has ON(account.login = has.account_login AND account.password = has.account_password)
JOIN game ON(game.name = has.game_name AND game.release_date = has.game_release_date)
WHERE (account.nickname = 'interdum');
```

```
    nickname                  name
    character varying (50)    character varying (50)
29  blandit                   Treeflex
30  blandit                   Ronstring
31  blandit                   Konklux
32  blandit                   Redhold
33  blandit                   Zaam-Dox
34  interdum                  Viva
35  interdum                  Andalax
36  interdum                  Konklux
37  interdum                  Veribet
38  interdum                  Andalax
```

```
/* get all the offices that don't have parking */

SELECT office.country, office.city, office.street, office.parking, studiokeep.name
FROM office
JOIN (SELECT *
FROM keep_office AS keep
JOIN game_studio ON (keep.game_studio = game_studio.name)) AS studiokeep
ON (office.country = studiokeep.office_country AND office.city = studiokeep.office_city AND office.street =
studiokeep.office_street)
EXCEPT
SELECT office.country, office.city, office.street, office.parking, game_studio.name
FROM office
JOIN keep_office AS keep ON (office.country = keep.office_country AND office.city = keep.office_city AND
office.street = keep.office_street)
JOIN game_studio ON (keep.game_studio = game_studio.name)
where (parking = true);
```

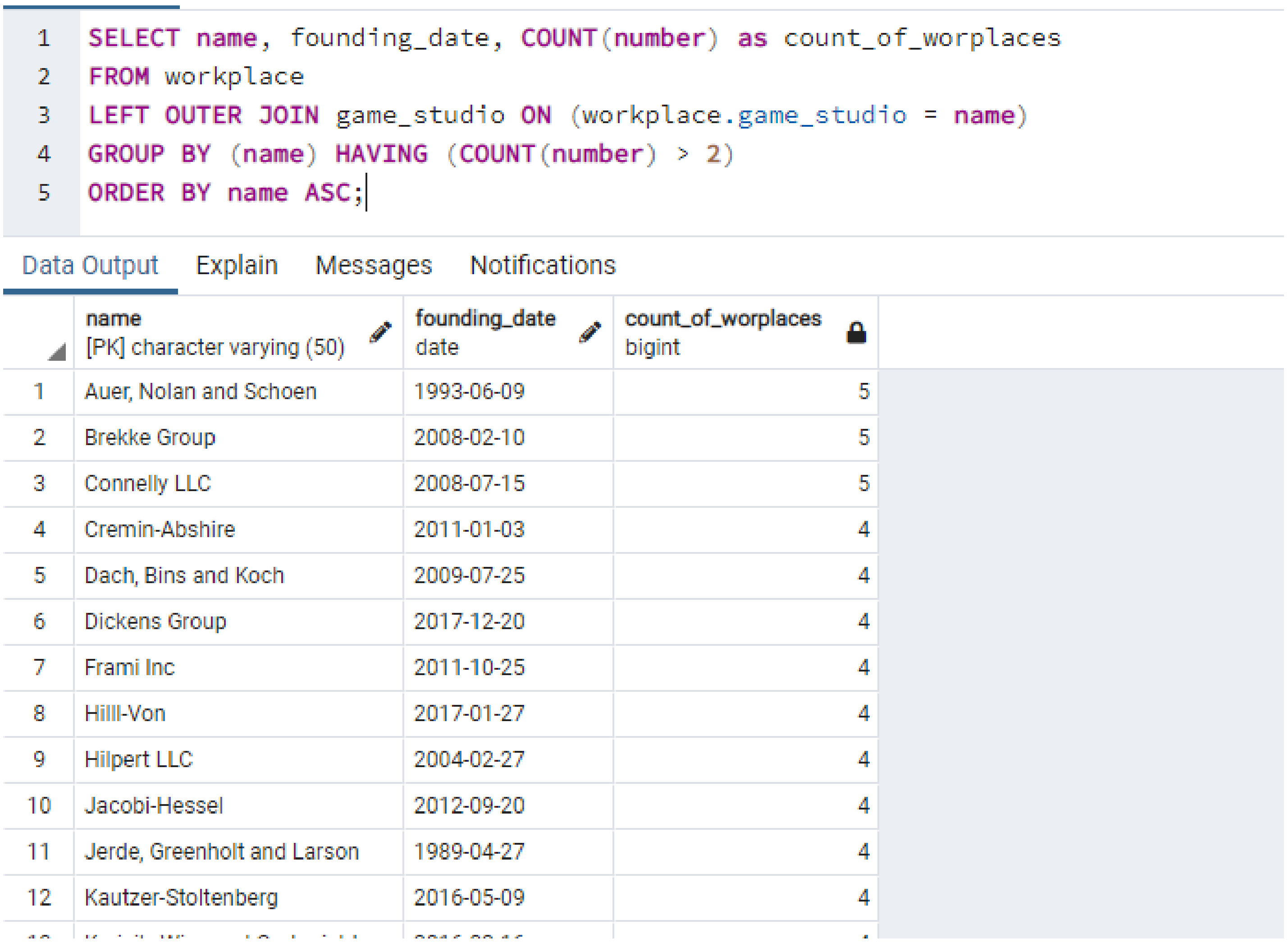


```
SELECT office.country, office.city, office.street, office.parking, studiokeep.name
FROM office
JOIN (SELECT *
FROM keep_office AS keep
JOIN game_studio ON (keep.game_studio = game_studio.name)) AS studiokeep
ON (office.country = studiokeep.office_country AND office.city = studiokeep.office_city AND office.street = studiokeep.office_street)
EXCEPT
SELECT office.country, office.city, office.street, office.parking, game_studio.name
FROM office
JOIN keep_office AS keep ON (office.country = keep.office_country AND office.city = keep.office_city AND office.street = keep.office_street)
JOIN game_studio ON (keep.game_studio = game_studio.name)
where (parking = true);
```

	country character varying (50)	city character varying (50)	street character varying (50)	parking boolean	name character varying (50)
1	Russia	Gayny	Talmadge	false	Toy Inc
2	Bolivia	Camargo	Nevada	false	Jacobi-Hessel
3	Indonesia	Tegalsari	Superior	false	Frami Inc
4	China	Binjiang	Norway Maple	false	Kuhn Group
5	China	Ningdun	Harbort	false	Raynor LLC
6	Colombia	Santa Lucía	Carey	false	Walsh LLC
7	Mexico	Morelos	Trailsway	false	Krajcik, Wiza and Cartwright
8	Sweden	Åtvidaberg	Warner	false	Dach, Bins and Koch
9	Russia	Charyshskoye	Bunting	false	Raynor LLC
10	Senegal	Thiès Nones	Fairfield	false	Brekke Group
11	China	Meicheng	Farwell	false	Pouros-Ruecker
12	Burkina Faso	Diapaga	Fair Oaks	false	Krajcik, Wiza and Cartwright
13	France	Annecy	Lillian	false	Watsica-Homenick
14	Indonesia	Cisewu	Macpherson	false	Brekke Group
15	Philippines	Apitong	Mosinee	false	Leannon LLC
16	Brazil	Cordeiro	Bay	false	Kautzer-Stoltenberg
17	Lithuania	Rietavas	Rieder	false	Jerde, Greenholt and Larson

```sql
/* get the number of workplaces in each game studio */

SELECT name, founding_date, COUNT(number) as count_of_worplaces
FROM workplace
LEFT OUTER JOIN game_studio ON (workplace.game_studio = name)
GROUP BY (name) HAVING (COUNT(number) > 2)
ORDER BY name ASC;
```



```sql
SELECT name, founding_date, COUNT(number) as count_of_worplaces
FROM workplace
LEFT OUTER JOIN game_studio ON (workplace.game_studio = name)
GROUP BY (name) HAVING (COUNT(number) > 2)
ORDER BY name ASC;
```

	name [PK] character varying (50)	founding_date date	count_of_worplaces bigint
1	Auer, Nolan and Schoen	1993-06-09	5
2	Brekke Group	2008-02-10	5
3	Connelly LLC	2008-07-15	5
4	Cremin-Abshire	2011-01-03	4
5	Dach, Bins and Koch	2009-07-25	4
6	Dickens Group	2017-12-20	4
7	Frami Inc	2011-10-25	4
8	Hilll-Von	2017-01-27	4
9	Hilpert LLC	2004-02-27	4
10	Jacobi-Hessel	2012-09-20	4
11	Jerde, Greenholt and Larson	1989-04-27	4
12	Kautzer-Stoltenberg	2016-05-09	4
[illegible]	[illegible]	[illegible]	[illegible]