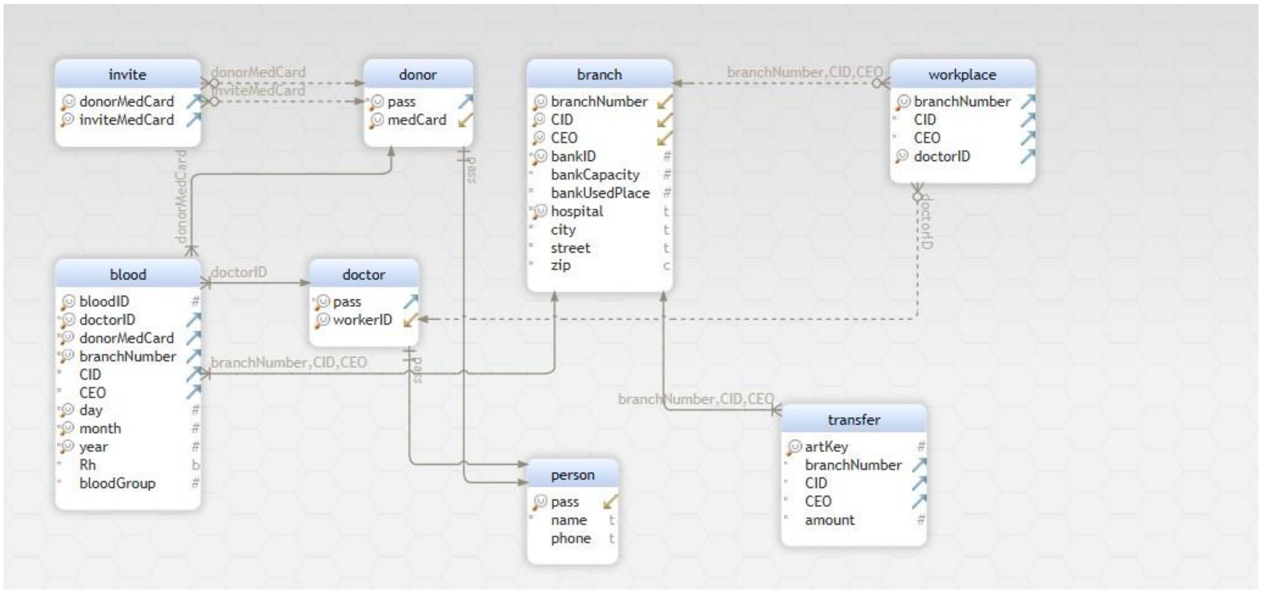
ER:



```
CREATE TABLE
    branch (
        branchNumber INTEGER,
        CID INTEGER,
        CEO VARCHAR(100),
        bankID INTEGER UNIQUE NOT NULL,
        bankCapacity INTEGER NOT NULL,
        bankUsedPlace INTEGER NOT NULL,
        hospital VARCHAR(100) UNIQUE NOT NULL,
        city VARCHAR(100) NOT NULL,
        street VARCHAR(100) NOT NULL,
        zip CHAR(20) NOT NULL,
        PRIMARY KEY (branchNumber, CID, CEO),
        CONSTRAINT bank_ch_space CHECK (bankUsedPlace <= bankUsedPlace)
    );

CREATE TABLE
    person (
        pass VARCHAR(10) PRIMARY KEY,
        NAME VARCHAR(100) NOT NULL,
        phone VARCHAR(100)
    );
```

```
CREATE TABLE
    donor (
        pass VARCHAR(10) NOT NULL UNIQUE,
        medCard VARCHAR(12),
        PRIMARY KEY (medCard),
        CONSTRAINT pass_fk FOREIGN KEY (pass) REFERENCES person (pass)
        ON
        UPDATE CASCADE
            ON
        DELETE CASCADE
    );

CREATE TABLE
    doctor (
        pass VARCHAR(10) NOT NULL UNIQUE,
        workerID INTEGER,
        PRIMARY KEY (workerID),
        CONSTRAINT pass_fk FOREIGN KEY (pass) REFERENCES person (pass)
        ON
        UPDATE CASCADE
            ON
        DELETE CASCADE
    );

CREATE TABLE
    invite (
        donorMedCard VARCHAR(20),
        inviteMedCard VARCHAR(20),
        PRIMARY KEY (donorMedCard, inviteMedCard),
        CONSTRAINT donor_fk FOREIGN KEY (donorMedCard) REFERENCES donor(medCard)
        ON
        UPDATE CASCADE
            ON
        DELETE
            CASCADE,
            CONSTRAINT invited_fk FOREIGN KEY (inviteMedCard) REFERENCES donor(medCard)
            ON
        UPDATE CASCADE
            ON
        DELETE CASCADE
    );
```

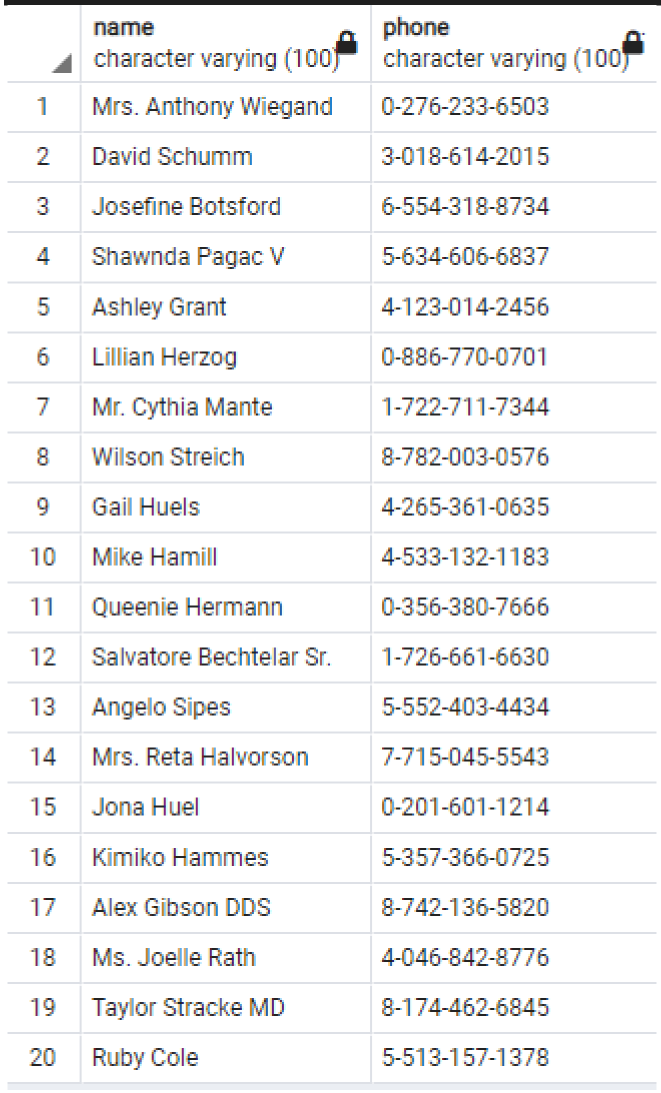
```sql
CREATE TABLE
    transfer (
        artKey INTEGER,
        branchNumber INTEGER NOT NULL,
        CID INTEGER NOT NULL,
        CEO VARCHAR(100) NOT NULL,
        amount INTEGER NOT NULL,
        PRIMARY KEY (artKey),
        CONSTRAINT branch_fk FOREIGN KEY (branchNumber, CID, CEO) REFERENCES branch (branchNumber, CID, CEO)
        ON
        UPDATE CASCADE
            ON
        DELETE
            CASCADE,
            CONSTRAINT amount_ch CHECK (amount >= 0)
    );

CREATE TABLE
    workplace (
        branchNumber INTEGER,
        CID INTEGER NOT NULL,
        CEO VARCHAR(100) NOT NULL,
        doctorID INTEGER,
        PRIMARY KEY (branchNumber, doctorID),
        CONSTRAINT branch_fk FOREIGN KEY (branchNumber, CID, CEO) REFERENCES branch (branchNumber, CID, CEO)
        ON
        UPDATE CASCADE
            ON
        DELETE
            CASCADE,
            CONSTRAINT doctor_fk FOREIGN KEY (doctorID) REFERENCES doctor(workerID)
            ON
        UPDATE CASCADE
            ON
        DELETE CASCADE
    );
```

```sql
CREATE TABLE
    blood (
        bloodID INTEGER,
        doctorID INTEGER NOT NULL,
        donorMedCard VARCHAR(12) NOT NULL,
        branchNumber INTEGER NOT NULL,
        CID INTEGER NOT NULL,
        CEO VARCHAR(100) NOT NULL,
        DAY SMALLINT NOT NULL,
        MONTH SMALLINT NOT NULL,
        YEAR SMALLINT NOT NULL,
        Rh BOOLEAN NOT NULL,
        bloodGroup SMALLINT NOT NULL,
        PRIMARY KEY (bloodID),
        CONSTRAINT blood_uq UNIQUE (
            doctorID,
            donorMedCard,
            branchNumber,
            DAY,
            MONTH,
            YEAR
        ),
        CONSTRAINT doctor_fk FOREIGN KEY (doctorID) REFERENCES doctor (workerID)
        ON
        UPDATE CASCADE
            ON
        DELETE
            CASCADE,
            CONSTRAINT donor_fk FOREIGN KEY (donorMedCard) REFERENCES donor (medCard)
            ON
        UPDATE CASCADE
            ON
        DELETE
            CASCADE,
            CONSTRAINT branch_fk FOREIGN KEY (branchNumber, CID, CEO) REFERENCES branch (branchNumber, CID, CEO)
            ON
        UPDATE CASCADE
            ON
        DELETE CASCADE
    );
```

QUERIES:

1.

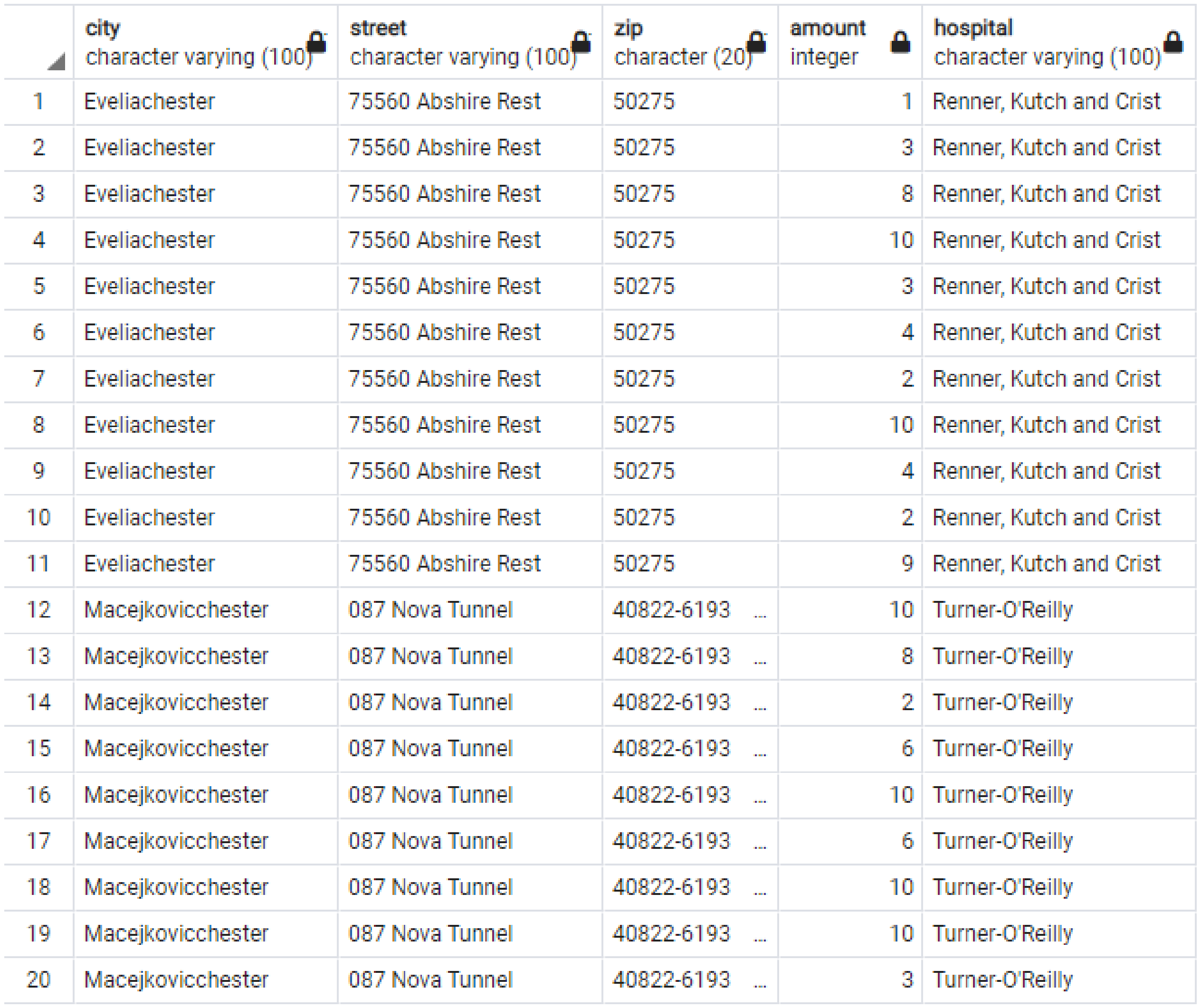
```
--INNER JOIN
--Vypisuje jmena lekaru, kteri maji telefonni cislo
SELECT name, phone
    FROM person
    JOIN doctor
        ON ((doctor.pass = person.pass) AND (person.phone is not null))
LIMIT(50);
```



	name character varying (100)	phone character varying (100)
1	Mrs. Anthony Wiegand	0-276-233-6503
2	David Schumm	3-018-614-2015
3	Josefine Botsford	6-554-318-8734
4	Shawnda Pagac V	5-634-606-6837
5	Ashley Grant	4-123-014-2456
6	Lillian Herzog	0-886-770-0701
7	Mr. Cythia Mante	1-722-711-7344
8	Wilson Streich	8-782-003-0576
9	Gail Huels	4-265-361-0635
10	Mike Hamill	4-533-132-1183
11	Queenie Hermann	0-356-380-7666
12	Salvatore Bechtelar Sr.	1-726-661-6630
13	Angelo Sipes	5-552-403-4434
14	Mrs. Reta Halvorson	7-715-045-5543
15	Jona Huel	0-201-601-1214
16	Kimiko Hammes	5-357-366-0725
17	Alex Gibson DDS	8-742-136-5820
18	Ms. Joelle Rath	4-046-842-8776
19	Taylor Stracke MD	8-174-462-6845
20	Ruby Cole	5-513-157-1378

2.

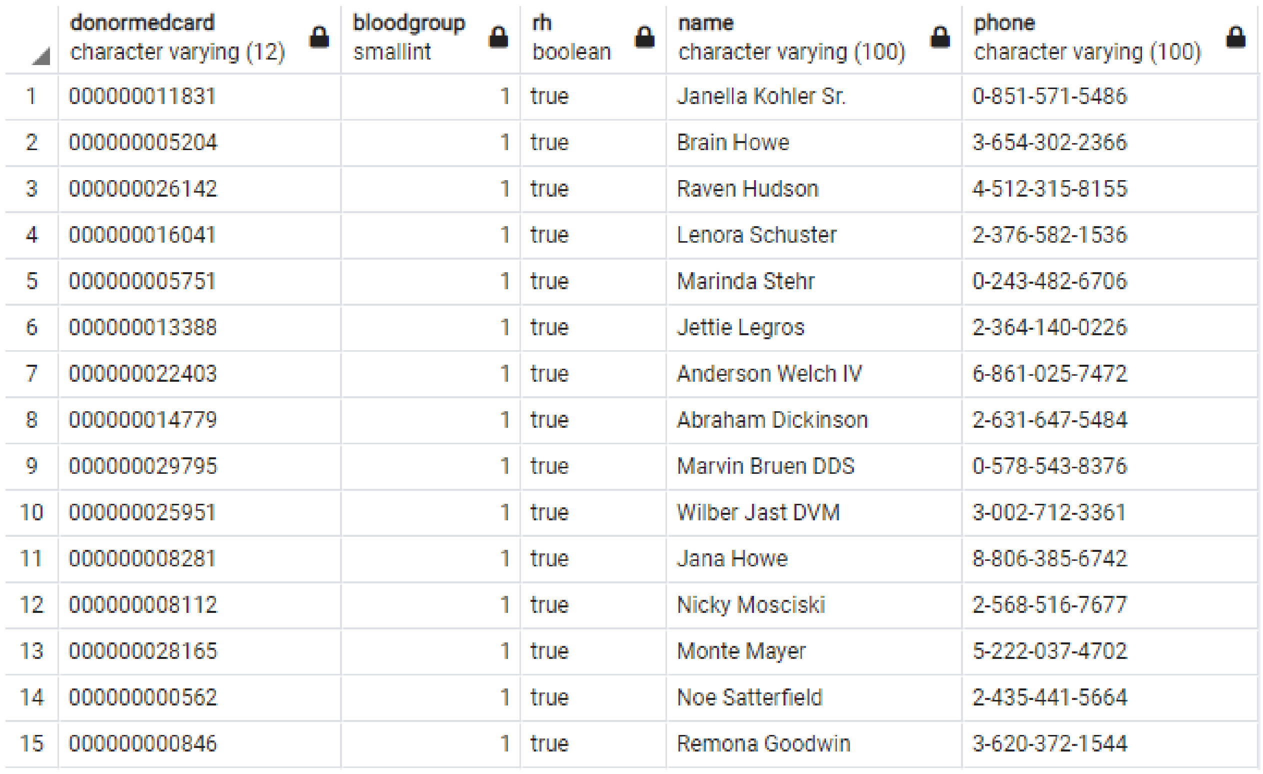
```sql
--OUTER JOIN
--Shows the address of branch, amount of transported blood and the name of the
hospital, to which blood was transported
SELECT city, street, zip, amount, hospital
FROM branch
    LEFT OUTER JOIN transfer
    ON (transfer.branchnumber = branch.branchnumber)
LIMIT(20);
```



	city character varying (100)	street character varying (100)	zip character (20)	amount integer	hospital character varying (100)
1	Eveliachester	75560 Abshire Rest	50275	1	Renner, Kutch and Crist
2	Eveliachester	75560 Abshire Rest	50275	3	Renner, Kutch and Crist
3	Eveliachester	75560 Abshire Rest	50275	8	Renner, Kutch and Crist
4	Eveliachester	75560 Abshire Rest	50275	10	Renner, Kutch and Crist
5	Eveliachester	75560 Abshire Rest	50275	3	Renner, Kutch and Crist
6	Eveliachester	75560 Abshire Rest	50275	4	Renner, Kutch and Crist
7	Eveliachester	75560 Abshire Rest	50275	2	Renner, Kutch and Crist
8	Eveliachester	75560 Abshire Rest	50275	10	Renner, Kutch and Crist
9	Eveliachester	75560 Abshire Rest	50275	4	Renner, Kutch and Crist
10	Eveliachester	75560 Abshire Rest	50275	2	Renner, Kutch and Crist
11	Eveliachester	75560 Abshire Rest	50275	9	Renner, Kutch and Crist
12	Macejkovicchester	087 Nova Tunnel	40822-6193 ...	10	Turner-O'Reilly
13	Macejkovicchester	087 Nova Tunnel	40822-6193 ...	8	Turner-O'Reilly
14	Macejkovicchester	087 Nova Tunnel	40822-6193 ...	2	Turner-O'Reilly
15	Macejkovicchester	087 Nova Tunnel	40822-6193 ...	6	Turner-O'Reilly
16	Macejkovicchester	087 Nova Tunnel	40822-6193 ...	10	Turner-O'Reilly
17	Macejkovicchester	087 Nova Tunnel	40822-6193 ...	6	Turner-O'Reilly
18	Macejkovicchester	087 Nova Tunnel	40822-6193 ...	10	Turner-O'Reilly
19	Macejkovicchester	087 Nova Tunnel	40822-6193 ...	10	Turner-O'Reilly
20	Macejkovicchester	087 Nova Tunnel	40822-6193 ...	3	Turner-O'Reilly

3.

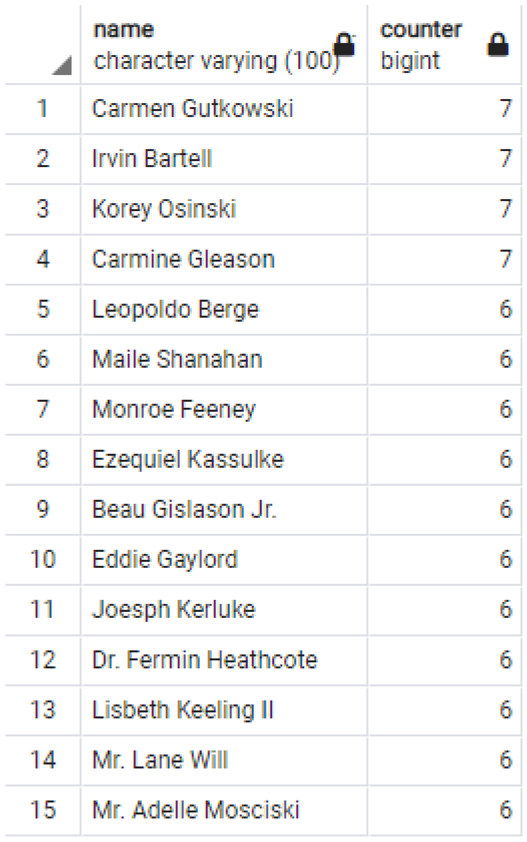
```sql
--Data condition
--Shows donor's info (name, phone if has one, med. card), who has donated blood
of first group and positive Rh
SELECT donormedcard, bloodGroup, Rh, NAME, phone
FROM blood
    JOIN donor
    ON (blood.donormedcard = donor.medcard)
    JOIN person
    ON (person.pass = donor.pass)
WHERE (bloodGroup = 1) AND (Rh = TRUE) AND (phone IS NOT NULL)
LIMIT(20)
OFFSET 2;
```



	donormedcard character varying (12)	bloodgroup smallint	rh boolean	name character varying (100)	phone character varying (100)
1	000000011831	1	true	Janella Kohler Sr.	0-851-571-5486
2	000000005204	1	true	Brain Howe	3-654-302-2366
3	000000026142	1	true	Raven Hudson	4-512-315-8155
4	000000016041	1	true	Lenora Schuster	2-376-582-1536
5	000000005751	1	true	Marinda Stehr	0-243-482-6706
6	000000013388	1	true	Jettie Legros	2-364-140-0226
7	000000022403	1	true	Anderson Welch IV	6-861-025-7472
8	000000014779	1	true	Abraham Dickinson	2-631-647-5484
9	000000029795	1	true	Marvin Bruen DDS	0-578-543-8376
10	000000025951	1	true	Wilber Jast DVM	3-002-712-3361
11	000000008281	1	true	Jana Howe	8-806-385-6742
12	000000008112	1	true	Nicky Mosciski	2-568-516-7677
13	000000028165	1	true	Monte Mayer	5-222-037-4702
14	000000000562	1	true	Noe Satterfield	2-435-441-5664
15	000000000846	1	true	Remona Goodwin	3-620-372-1544

4.

```
--Agregation, condition on agregation function, sorting a paging
--Shows how many times each donor has donated a blood, sorted in descending order
SELECT NAME, COUNT(*) AS counter
FROM blood
    JOIN donor
    ON (blood.donormedcard = donor.medcard)
    JOIN person
    ON (donor.pass = person.pass)
GROUP BY (NAME)
HAVING (NAME NOT LIKE 'A%')
ORDER BY (counter) DESC
LIMIT(15)
```



	name character varying (100)	counter bigint
1	Carmen Gutkowski	7
2	Irvin Bartell	7
3	Korey Osinski	7
4	Carmine Gleason	7
5	Leopoldo Berge	6
6	Maile Shanahan	6
7	Monroe Feeney	6
8	Ezequiel Kassulke	6
9	Beau Gislason Jr.	6
10	Eddie Gaylord	6
11	Joesph Kerluke	6
12	Dr. Fermin Heathcote	6
13	Lisbeth Keeling II	6
14	Mr. Lane Will	6
15	Mr. Adelle Mosciski	6