Name

CWID

Homework Assignment 2

Due Date: October 20th, 2022

CS425 - Database Organization Results

Please leave this empty!

2.1 2.2

2.3 2.4 2.5 2.6 2.7 2.8 2.9 2.10

2.11 2.12

2.15 2.16 2.17 2.18 2.19 Sum

Instructions

- Try to answer all the questions using what you have learned in class
- **When writing a query, write the query in a way that it would work over all possible database instances and not just for the given example instance!**
- Some questions are marked as bonus. You do not have to answer these questions to get full points for the assignment. However, you can get bonus points for these questions!
- **Please submit the homework electronically using blackboard**
- **The due date is 10/20!**

Consider the following database schema and example instance storing information about theatre plays:

play

title	written	genre
Look Back in Anger	1956-05-08	realist
The Entertainer	1957-04-10	realist
Oresteia	458-01-01 BC	classical
The Suppliants	455-01-01 BC	classical
A Midsummer Nights Dream	1605-01-01	comedy
Macbeth	1623-01-01	tragedy

writer

name	born
William Shakespeare	1564-01-01
Aeschylus	525-01-01 BC
John Osborne	1929-12-12

authorship

writer	play	written
John Osborne	Look Back in Anger	1956-05-08
John Osborne	The Entertainer	1957-04-10
Aeschylus	Oresteia	458-01-01 BC
Aeschylus	The Suppliants	455-01-01 BC
William Shakespeare	A Midsummer Nights Dream	1605-01-01
William Shakespeare	Macbeth	1623-01-01

actor

name	born
Thespis	510-01-01 BC
John Malkovich	1953-12-09
Frances McDormand	1957-06-23

character

name	play	written
Duncan	Macbeth	1623-01-01
Hecate	Macbeth	1623-01-01
Lady Macbeth	Macbeth	1623-01-01
Jimmy	Look Back in Anger	1956-05-08
Billy Rice	The Entertainer	1957-04-10
Danaus	The Suppliants	455-01-01 BC
Pelasgus	The Suppliants	455-01-01 BC

scene

play	written	nr	title	description
The Entertainer	1957-04-10	1	Act 1	Billy Rice a retired music-hall star ...
A Midsummer Nights Dream	1605-01-01	1	Act 1 Scene 1	Theseus and Hyppolyta who are four days ...

performance

play	written	pdate	theatre
Look Back in Anger	1956-05-08	2022-11-08	Steppenwolf Theatre
Look Back in Anger	1956-05-08	2022-11-15	Steppenwolf Theatre
The Suppliants	455-01-01 BC	2022-12-05	Navy Pier

performs

play	written	pdate	theatre	actor	character
Look Back in Anger	1956-05-08	2022-11-08	Steppenwolf Theatre	John Malkovich	Jimmy
Look Back in Anger	1956-05-08	2022-11-15	Steppenwolf Theatre	John Malkovich	Jimmy
The Suppliants	455-01-01 BC	2022-12-05	Navy Pier	Frances McDormand	Danaus

Hints:

- All the attributes that have integer values are of type `INT`; numbers with decimal point are of type `NUMERIC`; the attribute *written*, *born*, and *pdate* attributes are of type `DATE`; others are of type `VARCHAR`.
- Attributes with black background form the primary key of an relation.
- The attributes *play* and *written* of relations *authorship*, *character*, *scene*, and *performance* are foreign keys to relation *play*.
- The attributes *play*, *written*, *pdate*, and *theatre* of relation *performs* is a foreign key to relation *performance*, *play*, *written*, *character* are a foreign key to relation *character*, and *actor* is a foreign key to relation *actor*.
- The attribute *writer* of relation *authorship* is a foreign key to relation *writer*.

Part 2.1 SQL DDL (Total: 14 Points)

Question 2.1.1 (7 Points)

Write an SQL statement that adds a *length* attribute to relation *play* which stores the length in minutes of the play and adds length to the primary key for this table. Furthermore, enforce that the length of any play is always positive. The default value for length should be 60 minutes.

Solution

```
ALTER TABLE play
  ADD COLUMN length INT CHECK (length > 0) DEFAULT 60;

ALTER TABLE play DROP CONSTRAINT play_pkey CASCADE;
ALTER TABLE play ADD PRIMARY KEY (title, written, length);
```

Question 2.1.2 (7 Points)

Write an SQL statement that adds a constraint to the *performance* relation to make sure that the *pdate* attribute cannot be `NULL`, and that the value of this attribute is larger than `2022-01-01`. Furthermore, the default value for this attribute should be the current date. Use `CURRENT_DATE` to access the current date.

Solution

```
ALTER TABLE performance
  ALTER pdate SET NOT NULL,
  ALTER pdate SET DEFAULT CURRENT_DATE,
  ADD CONSTRAINT datecheck CHECK(pdate > '2022-01-01');
```

Part 2.2 SQL Queries (Total: 56 + 10 BONUS Points)

Question 2.2.1 (5 Points)

Write an SQL query that returns the `title` and `written` date of play which are performed at least 100 years after they where written. Do not return duplicates.

Solution

```
SELECT DISTINCT p.title, p.written
FROM play p, performance f
WHERE p.title = f.play
      AND p.written = f.written
      AND f.pdate - '100 years'::interval > p.written;
```

Natural join could be replaced by other correct forms, same below.

Question 2.2.2 (5 Points)

Write an SQL query that returns for each play and character in that play, the number of actors (do not double count actors which have played a character more than once) which have played that character.

Solution

```
SELECT c.name, c.play, c.written, count(DISTINCT actor)
FROM character c, performs p
WHERE c.name = p.character AND c.play = p.play AND c.written = p.written
GROUP BY c.name, c.play, c.written;
```

Question 2.2.3 (7 Points)

Write an SQL query that returns actors and theatre pairs (A, T) if actor A has played in at least half of all the performances at theatre T.

Solution

```
SELECT theatre, actor
FROM performs p1
GROUP BY theatre, actor
HAVING count(*) >
       0.5 * (SELECT count(*)
              FROM performs p2
              WHERE p1.theatre = p2.theatre);
```

Question 2.2.4 (7 Points)

Write an SQL query that returns for each writer the number of actors that have never played a character from any of the plays of the writer.

Solution

```
SELECT w.name, count(*)
  FROM writer w, actor a
 WHERE NOT EXISTS (SELECT *
                     FROM performs p NATURAL JOIN authorship s
                    WHERE p.actor = a.name
                          AND s.writer = w.name)
 GROUP BY w.name;
```

Question 2.2.5 (8 Points)

Write an SQL query that returns the name of writers that have only written plays with more than 10 scenes.

Solution

```
SELECT name
FROM writer w
WHERE NOT EXISTS (SELECT *
                    FROM authorship a
                   WHERE a.writer = w.name
                         AND 10 > (SELECT count(*)
                                     FROM scene s
                                    WHERE a.play = s.play AND a.written = s.written))
```

Question 2.2.6 (8 Points)

Write an SQL query that returns a rolling sum over the years of the number of plays that have been written up to that point in time. For instance for year 1623, when Macbeth was written you need to return the number of plays that have been written up to and including this year. Note that you can use `EXTRACT(YEAR FROM x)` to extract the year from a date.

Solution

```
SELECT EXTRACT(YEAR FROM written) AS year,
       count(*) OVER (ORDER BY EXTRACT(YEAR FROM written)
                      GROUPS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW)
FROM play;
```

Question 2.2.7 (8 Points)

Write an SQL query that returns the title of the play with the most scenes.

Solution

```
WITH scenecnt AS (
  SELECT count(*) AS numscene, play, written
  FROM scene
  GROUP BY play, written)
SELECT play, written
FROM scenecnt s1
WHERE numscene = (SELECT max(numscene) FROM scenecnt);
```

Question 2.2.8 (8 Points)

Write an SQL query that returns names of actors which played in a play written by a writer that was born more than 200 years before the actor was born.

Solution

```
SELECT name
FROM authorship a, writer w, performance p, performs m, actor c
WHERE a.writer = w.name
      AND a.play = p.play
      AND a.written = p.written
      AND p.play = m.play
      AND p.written = m.written
      AND m.actor = c.name
      AND w.born > w.born + '200 years'::interval;
```

Question 2.2.9 BONUS (5 Points)

Write an SQL query which returns for each genre (that occurs in table `play`), the first play that was written belonging to this genre. For instance, in our example database the first comedy is "*A Midsummer Night's Dream*".

Solution

```
SELECT *
FROM play p1
WHERE written = (SELECT min(written) FROM play p2 WHERE p1.genre = p2.genre);
```

Question 2.2.10 BONUS (5 Points)

Write an SQL query that returns the title of plays that have more scenes than characters.

Solution

```
SELECT title
FROM play p
WHERE (SELECT count(*)
         FROM scene s
        WHERE s.play = p.title
              AND s.written = p.written)
      > (SELECT count(*)
           FROM character c
          WHERE c.play = p.title
                AND c.written = p.written);
```

Part 2.3 SQL Updates (Total: 30 + 5 BONUS Points)

Question 2.3.1 (7 Points)

Delete all characters of plays that were written by writers born before year 1000.

Solution

```
DELETE FROM character
 WHERE (play,written) IN (SELECT a.play, a.written
                            FROM authorship a, writer w
                           WHERE a.writer = w.name AND w.born < '1000-01-01'::DATE);
```

Question 2.3.2 (8 Points)

Add a new character *King Kong* to play *Looking Back in Anger* and add a performance by actor *John Malkovich* for this role to the `2022-11-15` performance at *Steppenwolf Theatre* of this play.

Solution

```
INSERT INTO character VALUES ('King Kong', 'Look Back in Anger', '1956-05-08');
INSERT INTO performs
       VALUES ('Look Back in Anger',
               '1956-05-08',
               '2022-11-15',
               'Steppenwolf Theatre',
               'John Malkovich',
               'King Kong');
```

Question 2.3.3 (6 Points)

Actor John Malkovitch refuses to play *King Kong* in the newly inserted *2022-11-15* performance of *Look Back in Anger*. Change this performance to actor "*Frances McDormand*".

Solution

```
UPDATE performs SET actor = 'Frances McDormand'
 WHERE play = 'Look Back in Anger'
       AND written = '1956-05-08'
       AND pdate = '2022-11-15'
       AND theatre = 'Steppenwolf Theatre'
       AND character = 'King Kong';
```

Question 2.3.4 (9 Points)

Update the performance table such that all performances from *Steppenwolf Theatre* are moved to *Navy Pier* and all performance from *Navy Pier* are moved to *Theatre in the Park*. Note that we expect you to write a single statement that implements this.

Solution

```
UPDATE performance
   SET theatre = (CASE
                      WHEN theatre = 'Steppenwolf Theatre' THEN 'Navy Pier'
                      WHEN theatre = 'Navy Pier' THEN 'Theatre in the Park'
                      ELSE theatre
                  END);
```

Question 2.3.5 BONUS (5 Points)

Update the `pdate` by delaying it by one week (7 days) for all performances of plays written by *William Shakespeare*. Note that Postgres has an `interval` data type that can be used for arithmetic over dates, e.g., '2015−01−01::DATE␣+␣'1 day'::interval evaluates to `2015-01-02`.

Solution

```
UPDATE performance
SET pdate = pdate + '7␣days'::interval
WHERE (play, written) IN (SELECT play, written
                          FROM authorship
                          WHERE writer = 'William␣Shakespeare');
```