1. Orchestra Celebration

The New York Philharmonic is one of America's largest orchestras. In honor of its many famous musicians, the director is planning a special event. She wants to hold a concert to celebrate the top soloists from its history. You have been asked to determine which soloists should receive recognition.

The director has given you the following requirements:

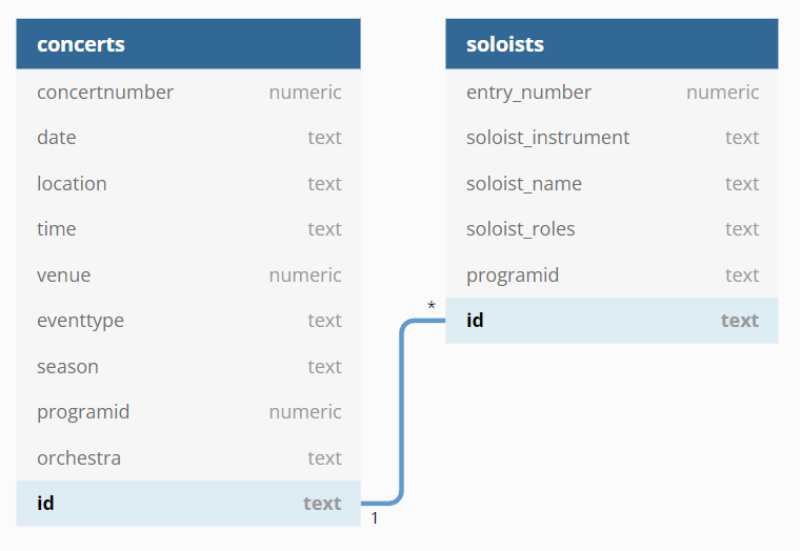
- The results should only include the top 1% of soloists calculated by the total number of concerts performed.
- Limit your results to entries where the orchestra is 'New York Philharmonic' and the event type is 'Subscription Season'. Soloist performances in other orchestras or event types should not count towards the calculation of the top soloists.
- The director is only interested in individual soloists. You will need to exclude all of the following soloists: 'Chorus', 'No Soloist', 'New York Choral Artists', and 'Schola Cantorum of NY'. You should also exclude any soloist with 'choir' in the name.

Your result should contain the following columns. It should meet all requirements as described.

column	requirements
name	The name of the soloist, with the first name followed by the last name (e.g., Jane Smith). Please note that in the soloists table, names are in reverse order (last name, first name).
first_date	The first date the soloist ever performed with the orchestra, in the format '01 Jan 2015' (i.e., day as an integer, short month name, and year as an integer).
last_date	The last date the soloist ever performed with the orchestra, in the format '01 Jan 2015' (i.e., day as an integer, short month name, and year as an integer).
total_concerts	The total number of concerts the soloist performed.

Order your results by the total number of concerts performed in descending order, and then by soloist name in alphabetical order. The data you will need is available in the two tables detailed in the schema below.

Database Schema



In [2]:

```
%%sql
postgresql:///orchestra

SELECT
    name,
    first_date,
    last_date,
    total_concerts
FROM
    (SELECT
        CONCAT(
            RIGHT(s.soloist_name, LENGTH(s.soloist_name)-STRPOS(s.soloist_name, ',' )),
            ' ',
            LEFT(s.soloist_name, STRPOS(s.soloist_name, ',' ) - 1)
        ) AS name,
        TO_CHAR(CAST(MIN(date) AS DATE), 'DD Mon YYYY') AS first_date,
        TO_CHAR(CAST(MAX(date) AS DATE), 'DD Mon YYYY') AS last_date,
        COUNT(*) AS total_concerts,
        PERCENT_RANK() OVER(ORDER BY COUNT(*) DESC) AS percent_rank

    FROM soloists AS s
    JOIN concerts AS c
        ON s.id = c.id
    WHERE c.orchestra = 'New York Philharmonic'
        AND c.eventtype = 'Subscription Season'
        AND soloist_name NOT IN
            ('Chorus',
             'No Soloist',
             'New York Choral Artists',
             'Schola Cantorum of NY')
        AND LOWER(soloist_name) NOT LIKE '%choir%'
    GROUP BY s.soloist_name

    ORDER BY total_concerts DESC, name) AS sub
WHERE percent_rank < 0.01
```

29 rows affected.

Out[2]:

name	first_date	last_date	total_concerts
Glenn Dicterow	16 Oct 1980	25 Jan 2014	277

name	first_date	last_date	total_concerts
Philip Smith	08 Nov 1979	17 Dec 2011	212
Philip Myers	24 Jan 1980	07 Nov 2015	193
Stanley Drucker	12 Oct 1961	09 Jun 2009	186
John Corigliano	17 Dec 1921	18 Apr 1966	172
Lorne Munroe	22 Oct 1964	03 Feb 1996	145
Carter Brey	27 Nov 1996	08 Jun 2017	129
Thomas Stacy	19 Apr 1973	14 Mar 2009	129
The Camerata Singers	09 Apr 1964	14 Sep 1977	123
Joseph Robinson	30 Nov 1978	25 Jan 2003	119
Pinchas Zukerman	05 Feb 1969	12 Jan 2013	110
Scipione Guidi	11 Dec 1921	12 Mar 1931	101
Emanuel Ax	29 Sep 1977	07 Jan 2017	99
Rudolf Serkin	20 Feb 1936	14 Sep 1983	99
Judith LeClair	11 Mar 1982	07 Nov 2015	95
Mishel Piastro	18 Nov 1931	15 Jan 1956	92
Isaac Stern	07 Dec 1944	10 Dec 1992	91
Harold Gomberg	06 Apr 1944	19 Dec 1981	88
Alicia de Larrocha	29 Dec 1965	18 Oct 2003	85
Yefim Bronfman	21 Sep 1978	28 Jan 2017	82
Zino Francescatti	19 Nov 1939	16 Dec 1975	82
Rudolf Firkusny	28 Oct 1943	29 Oct 1991	80
Robert Casadesus	20 Jan 1935	01 Dec 1969	79
John Wummer	15 Apr 1943	30 May 1965	76
Andre (André) Watts	31 Jan 1963	11 Dec 2012	75
Charles Rex	16 Feb 1981	17 Feb 1998	72
Cynthia Phelps	10 Dec 1992	08 Jun 2017	71
Nathan Milstein	23 Jan 1930	23 Oct 1982	67
Itzhak Perlman	09 May 1965	28 Sep 2010	64

In [3]:

```
%%nose
last_output = _
import pandas as pd
pd.set_option("max_colwidth", 5000)
import numpy as np
import re
from pandas.api.types import is_numeric_dtype
```

```
student_result = last_output.DataFrame()

# True solution
names = ['Glenn Dicterow', 'Philip Smith', 'Philip Myers', 'Stanley Drucker', 'John Cor
first_date = ['16 Oct 1980', '08 Nov 1979', '24 Jan 1980', '12 Oct 1961', '17 Dec 1921'
last_date = ['25 Jan 2014', '17 Dec 2011', '07 Nov 2015', '09 Jun 2009', '18 Apr 1966',
number = [277, 212, 193, 186, 172, 145, 129, 129, 123, 119, 110, 101, 99, 99, 95, 92, 9
test_solution = pd.DataFrame({"name": names, "first_date": first_date, "last_date": las

# Alt solution for people who missed updated instructions
name = ['Glenn Dicterow', 'Philip Smith', 'Philip Myers', 'Stanley Drucker', 'John Cori
first_date = ['16 Oct 1980', '08 Nov 1979', '24 Jan 1980', '12 Oct 1961', '17 Dec 1921'
last_date = ['25 Jan 2014', '17 Dec 2011', '07 Nov 2015', '09 Jun 2009', '18 Apr 1966',
total_concerts = [277, 212, 193, 186, 172, 145, 129, 129, 119, 110, 101, 99, 99, 95, 92
alt_test_solution = pd.DataFrame({"name": name, "first_date": first_date, "last_date":

student_result_trimmed = student_result[0:28]
test_solution_trimmed = test_solution[0:28]
alt_test_solution_trimmed = alt_test_solution[0:28]

try:
    student_result_trimmed['name'] = student_result_trimmed['name'].str.strip()
    test_solution_trimmed['name'] = test_solution_trimmed['name'].str.strip()
    alt_test_solution_trimmed['name'] = alt_test_solution_trimmed['name'].str.strip()
except:
    None

def test_project():
    test_solution.columns = [x.lower() for x in test_solution.columns.tolist()]
    student_result.columns = [x.lower() for x in student_result.columns.tolist()]

    # For each column in the DataFrame, ensure it's in the test_solution DataFrame
    assert len(student_result_trimmed) == 28, \
    'Your SQL ResultSet does not contain the expected number of rows!'
    assert all([x in test_solution_trimmed.columns for x in student_result_trimmed.colu
    'Your SQL ResultSet is missing the required columns!'
    # Check whether the values are correct
    # Alternatively, check result of people who missed the instructions
    for col in test_solution_trimmed.columns.tolist():
            assert (test_solution_trimmed[col].values == student_result_trimmed[col].va
             (alt_test_solution_trimmed[col].values == student_result_trimmed[col].valu
            'The values in your SQL ResultSet do not match the solution!'
```

Out[3]: 1/1 tests passed