CSC 370 - SUMMER 2020
DATABASE SYSTEMS
ASSIGNMENT 4
UNIVERSITY OF VICTORIA

Due: Thursday, July 9th, 2020 at 11:55pm. **Late assignments will not be accepted.**

This assignment will be accepted electronically, and will be marked using the same schema as the previous assignment. See the 'Submission and Evaluation' section below for details on the submission process and expected formatting of your answers. For all of the questions below, your answer must be **one** SQL query (including a terminating semicolon) which runs without errors on the `studdb1.csc.uvic.ca` or `studdb2.csc.uvic.ca` PostgreSQL database servers. Note that timeout errors (in which the server terminates your query for exceeding the maximum execution time) are considered errors. Queries which have errors will receive a mark of zero. All queries without errors will be marked out of two, with full marks given only to queries which produce the correct output and contain no assumptions besides the data given in the question (see the advice sections of assignment 2 for more details).

Some of the queries in this assignment involve floating point data. In the expected output shown below, there might be slight differences in floating point values due to rounding and formatting issues. You may assume that floating point values in your query results match the expected output if they are equivalent to four decimal places (e.g. if the expected output is written as '0.1234', the values '0.123398' and '0.1234567' would both be considered correct, but the value '0.123' would not).

Question 1: IMDB Queries [4 marks]
Create queries for each of the data retrieval problems below, using the `imdb` database. In the questions below, any reference to 'films' refers to titles with `title_type = 'movie'`.

(a) For each year between 2000 and 2017 (inclusive), list the primary name, production year, rating and number of votes of the film or films which attained the highest rating among all movies produced in that year which received at least 10000 votes. Both the rating and number of votes are stored in the `ratings` table.

Expected Query Result			
primary_name	year	rating	votes
Memento	2000	8.5000	942432
Gladiator	2000	8.5000	1095891
The Lord of the Rings: The Fellowship of the Ring	2001	8.8000	1370113
The Lord of the Rings: The Two Towers	2002	8.7000	1221886
The Lord of the Rings: The Return of the King	2003	8.9000	1349934
Anbe Sivam	2003	8.9000	10116
Black Friday	2004	8.6000	13513
Earthlings	2005	8.7000	14649
The Prestige	2006	8.5000	959259
The Lives of Others	2006	8.5000	290278
The Departed	2006	8.5000	975612
Like Stars on Earth	2007	8.5000	113421
The Dark Knight	2008	9.0000	1864795
Home	2009	8.6000	19621
Inception	2010	8.8000	1653611
The Intouchables	2011	8.6000	591006
The Dark Knight Rises	2012	8.4000	1269644
Django Unchained	2012	8.4000	1087769
CM101MMXI Fundamentals	2013	9.3000	38106
Interstellar	2014	8.6000	1120400
RangiTaranga	2015	8.7000	10286
The Mountain II	2016	9.6000	93071
Ayla: The Daughter of War	2017	9.1000	15807

(b) Select the primary name and episode count of each TV series (contained in the `tv_series` table) for which at least 6000 episodes have been produced.

Expected Query Result	
series_name	episode_count
Days of Our Lives	10240
Ohayou Tokushima	9502
Coronation Street	9316
Neighbours	8911
Six O'Clock News	8646
The Price Is Right	8461
One O'Clock News	8100
Jeopardy!	7599
EastEnders	7393
The Bold and the Beautiful	7236
Home and Away	7081
Wheel of Fortune	6564
Gute Zeiten, schlechte Zeiten	6413
The Tonight Show Starring Johnny Carson	6037
The Young and the Restless	6024

Question 2: BC Ferries Queries [14 marks]
The queries you write below should work correctly on any of the BC Ferries databases (`ferries_1month`, `ferries_3months`, `ferries_6months`, `ferries_9months`, `ferries_1year` or `ferries_3years`). For comparison, sample output is shown for both `ferries_1month` and `ferries_3years`.

The following definitions apply to all of the parts below unless otherwise stated.

- The 'duration' of a sailing is defined to be the number of minutes between the scheduled departure (not the actual departure) and the arrival of that sailing.
- Any reference to the 'day of a sailing' refers to the calendar day of the scheduled departure (not the actual departure) of that sailing.

(a) For many routes, there are simultaneous sailings in both directions at the same time. For example, on a typical day at 9:00am, a ferry leaves Tsawwassen for Swartz Bay and a different ferry leaves Swartz Bay for Tsawwassen. Not all routes or sailings have this property. We will define two vessels as 'paired up' if they have both served the same route number at the same (scheduled) departure time/date. Construct a query to count the number of times each distinct pair of ferries have been paired up. Your result must not contain counts for pairs of vessels which have never been paired up. Each distinct pair of ferries should appear only once in your result, and in the result rows, the vessel names of each pair must be ordered alphabetically (so the alphabetically lowest vessel name will be listed first).

Expected Query Result (`ferries_1month`)		
vessel1	vessel2	num_pairings
Coastal Inspiration	Queen of Alberni	158
Spirit of British Columbia	Spirit of Vancouver Island	10

Expected Query Result (`ferries_3years`)		
`vessel1`	`vessel2`	`num_pairings`
Coastal Inspiration	Queen of Alberni	5449
Spirit of British Columbia	Spirit of Vancouver Island	3552
Queen of Cowichan	Queen of Oak Bay	3048
Coastal Celebration	Queen of New Westminster	2261
Coastal Renaissance	Queen of Oak Bay	1825
Coastal Celebration	Spirit of British Columbia	1653
Coastal Celebration	Spirit of Vancouver Island	1247
Coastal Renaissance	Spirit of Vancouver Island	847
Coastal Renaissance	Queen of Alberni	814
Coastal Celebration	Coastal Renaissance	622
Coastal Renaissance	Queen of New Westminster	594
Coastal Inspiration	Queen of New Westminster	592
Queen of Coquitlam	Queen of Cowichan	548
Queen of Coquitlam	Queen of Oak Bay	481
Queen of Alberni	Queen of Coquitlam	400
Queen of Coquitlam	Queen of New Westminster	152
Coastal Renaissance	Spirit of British Columbia	53
Coastal Celebration	Coastal Inspiration	24
Queen of New Westminster	Spirit of Vancouver Island	17
Coastal Inspiration	Spirit of British Columbia	8
Queen of Coquitlam	Queen of Surrey	3
Coastal Renaissance	Queen of Cowichan	1

(b) The routes table contains the ‘nominal duration’ of each route, which is the expected crossing time. The nominal duration is determined by BC Ferries based on the average marine and traffic conditions, along with information like loading times and the speed of the vessels. We can test the accuracy of this calculation by computing the average time of each crossing. Construct a query to find, for each route number, the nominal duration (in minutes) and the average duration (in minutes) of a crossing based on all available data for that route. For this question, assume that the ‘duration’ of a particular sailing is the time between its scheduled departure and its arrival.

Expected Query Result (`ferries_1month`)		
`route_number`	`nominal_duration`	`avg_duration`
1	95	101.2424
3	40	44.9415
4	35	33.2479
8	20	20.5182
30	120	129.0709

Expected Query Result (ferries_3years)		
route_number	nominal_duration	avg_duration
1	95	92.8867
2	100	106.8488
3	40	45.1520
4	35	31.8939
8	20	21.9507
30	120	123.0126

(c) Suppose we define a sailing to be 'late' if the duration is at least five minutes longer[1] than the nominal duration in the routes table. Construct a query to find, for each month, the number of days for which at least one sailing occurred on Route 1 but no late sailings occurred on route number 1.

Expected Query Result (ferries_1month)	
month	count
4	1

Expected Query Result (ferries_3years)	
month	count
1	65
2	53
3	58
4	29
5	19
6	23
7	22
8	18
9	43
10	45
11	51
12	60

(d) Construct a query to find, for each vessel with any sailings, the total number of sailings it has made, the number of late sailings it has made (which may be zero) and the fraction of its sailings that were late (that is, the number of late sailings divided by the total number of sailings). You should be careful with this query: a vessel may be involved in multiple routes, each with a different nominal duration.

[1] A duration which is exactly five minutes longer is still considered late.

Expected Query Result (`ferries_1month`)

`vessel_name`	`total_sailings`	`late_sailings`	`late_fraction`
Coastal Inspiration	232	166	0.7155
Coastal Renaissance	39	14	0.3590
Queen of Alberni	163	43	0.2638
Queen of Capilano	440	30	0.0682
Queen of Coquitlam	68	16	0.2353
Queen of Surrey	274	100	0.3650
Skeena Queen	234	22	0.0940
Spirit of British Columbia	164	98	0.5976
Spirit of Vancouver Island	94	23	0.2447

Expected Query Result (`ferries_3years`)

`vessel_name`	`total_sailings`	`late_sailings`	`late_fraction`
Bowen Queen	1268	83	0.0655
Coastal Celebration	6506	406	0.0624
Coastal Inspiration	6545	2502	0.3823
Coastal Renaissance	5637	1220	0.2164
Mayne Queen	1	0	0.0000
Queen of Alberni	6903	1164	0.1686
Queen of Capilano	14277	2806	0.1965
Queen of Coquitlam	6650	2230	0.3353
Queen of Cowichan	7071	2946	0.4166
Queen of Cumberland	974	113	0.1160
Queen of New Westminster	3966	439	0.1107
Queen of Oak Bay	6609	4592	0.6948
Queen of Surrey	12759	4555	0.3570
Quinitsa	8	8	1.0000
Skeena Queen	7312	378	0.0517
Spirit of British Columbia	5539	1013	0.1829
Spirit of Vancouver Island	5916	791	0.1337

(e) Usually, you would expect that when the beginning of a sailing is delayed (that is, when `actual_departure` is much later than `scheduled_departure`), the arrival is also delayed and the sailing is late. However, in some cases, a vessel that departs late may still arrive on time. Define a 'made up sailing' to be any sailing which leaves at least 15 minutes after its scheduled departure but arrives less than (or equal to) five minutes late. Write a query to list the number of made up sailings in the dataset for each vessel. Only vessels with at least one made up sailing should be listed.

Expected Query Result (`ferries_1month`)

`vessel_name`	`made_up_sailings`
Coastal Inspiration	4
Coastal Renaissance	2
Spirit of British Columbia	3

Expected Query Result (`ferries_3years`)	
`vessel_name`	`made_up_sailings`
Coastal Celebration	109
Coastal Inspiration	70
Coastal Renaissance	53
Queen of Alberni	11
Queen of Coquitlam	7
Queen of Cowichan	1
Queen of New Westminster	28
Queen of Oak Bay	1
Skeena Queen	2
Spirit of British Columbia	57
Spirit of Vancouver Island	75

(f) Define a 'good day' for a particular route number to be a day where at least one sailing occurred and no late sailings occurred. For each route, find the maximum number of *good days* in a row over the entire dataset.

Note that the rather unimpressive results for the `ferries_1month` dataset compared to the other datasets seem to be the result of COVID-19 related service reductions and extra precautions (which may be increasing the number of late sailings).

Hint: This is probably easiest when you use a large number of CTE expressions (that is, subqueries in the `WITH` clause).

Expected Query Result (`ferries_1month`)	
`route_number`	`max_consecutive_good_days`
1	1
2	0
3	1
4	3
8	4
30	0

Expected Query Result (`ferries_3years`)	
`route_number`	`max_consecutive_good_days`
1	15
2	2
3	4
4	27
8	8
30	6

(g) For each route, output all date ranges where the maximum number of consecutive *good days* (as defined in part (f) above) was achieved. For some routes, only one date range will meet this criteria, but for others (e.g. route 2 in the `ferries_3years` dataset) there may be multiple date ranges of the same size. The columns containing the start and end dates should contain values of type `DATE` (created, for example, by the `make_date` function).

Expected Query Result (`ferries_1month`)			
`route_number`	`start_date`	`end_date`	`max_consecutive_good_days`
1	2020-04-26	2020-04-26	1
3	2020-04-15	2020-04-15	1
4	2020-04-19	2020-04-21	3
8	2020-04-29	2020-05-02	4

Expected Query Result (`ferries_3years`)			
`route_number`	`start_date`	`end_date`	`max_consecutive_good_days`
1	2017-12-21	2018-01-04	15
2	2017-05-20	2017-05-21	2
2	2017-09-26	2017-09-27	2
3	2018-02-24	2018-02-27	4
4	2018-09-02	2018-09-28	27
8	2019-03-06	2019-03-13	8
30	2017-09-23	2017-09-28	6

Question 3: VWSN Queries [10 marks]
Create queries for each of the data retrieval problems below, using the `vwsn_1year` database.

(a) Find the highest observed temperature in the dataset, along with the station number, station name and observation time of all cases where that temperature was reported.
Note: The expected output below is correct (that is, the observation shown is actually in the dataset, even though it appears to be unseasonably warm).

Expected Query Result			
`station_id`	`name`	`temperature`	`observation_time`
180	Captain Meares Elementary Secondary School	42.6000	2019-10-05 15:11:00

(b) For each station with station ID between 1 and 10 (inclusive), list the station ID, station name, maximum temperature observed at that station and observation time of **all** observations in the dataset in which the maximum temperature was attained at that station. Only include stations which actually have recorded observations in the dataset.

Expected Query Result			
station_id	name	max_temperature	observation_time
1	Ian Stewart Complex/Mt. Douglas High School	29.8000	2019-08-05 16:31:00
3	Strawberry Vale Elementary School	30.2000	2019-06-12 15:15:00
3	Strawberry Vale Elementary School	30.2000	2019-06-12 15:21:00
4	Oaklands Elementary School	30.3000	2019-06-12 14:46:00
4	Oaklands Elementary School	30.3000	2019-06-12 14:52:00
5	Cedar Hill Middle School	30.2000	2019-06-12 17:12:00
6	Marigold Elementary School/Spectrum High School	29.8000	2019-06-12 14:41:00
7	Campus View Elementary	29.6000	2019-06-12 17:22:00
8	Victoria High School	29.7000	2019-06-12 14:41:00
9	Frank Hobbs Elementary School	29.2000	2019-08-05 17:27:00
10	Macaulay Elementary School	27.3000	2019-06-12 12:26:00

(c) Find the IDs and names of all stations which have reported at least one observation at some point, but which did not report **any** observations in January 2020.

Expected Query Result	
station_id	name
13	Shoreline Middle School
16	Tillicum Elementary School
31	Sangster Elementary School
32	Colwood Elementary School
36	Hans Helgesen Elementary School
57	Wishart Elementary School
62	Deep Cove Elementary School
70	Parkland Secondary School
81	Braefoot Elementary School
94	Pender Islands Elementary and Secondary School
103	Frances Kelsey Secondary School
105	Port Renfrew Elementary School
108	Alberni Weather
109	Brentwood Elementary School
112	Saturna Elementary School
113	Mayne Island School
117	Saltspring Elementary and Middle Schools
124	Seaview Elementary School
131	Glenlyon Norfolk Junior School
133	Discovery Elementary School
136	Pleasant Valley Elementary School
160	Shawnigan Lake School
161	Bamfield Marine Sciences Centre
166	Maquinna Elementary School
180	Captain Meares Elementary Secondary School
195	North Island Distance Education School
196	Valley View Elementary School
226	Ditidaht Community School

(d) In this question (and the next question), define the ‘daily average temperature’ for a particular day to be the average of all observations from all stations on that day. Furthermore, for each month, define the ‘10 hottest days’ to be the top 10 days (by daily average temperature) and define the ‘10 coolest days’ to be the bottom 10 days (by daily average temperature). For each month/year pair in the dataset, compute the average daily average temperature across the ten hottest days and the average daily average temperature across the ten coolest days. Notice that you are computing an average of averages (first, find the average daily temperatures of each of the ten hottest days, then average those temperatures to produce the result). Hint: Use the rank() function with an appropriate over() clause (maybe in multiple places)

Expected Query Result			
year	month	hottest10_average	coolest10_average
2019	5	16.8834	12.0071
2019	6	17.8665	13.8339
2019	7	19.0044	15.9400
2019	8	19.7309	16.5326
2019	9	17.0929	12.3717
2019	10	10.8171	6.5475
2019	11	9.2537	3.6983
2019	12	7.3156	4.1378
2020	1	7.8368	0.9990
2020	2	6.3289	3.1054
2020	3	7.3842	3.7168
2020	4	11.5621	7.1426

(e) List the day, month and year (in separate columns) of all days whose daily average temperature (see previous question) was lower than the daily average temperature of **each** of the previous 28 days in the dataset. The result should not list any of the first 28 days in the dataset, since the metric cannot be computed for those days, but the data for those days should be used to compute the rest of the result. Hint: You may want to use both the min() and count() aggregation functions in combination with an over() clause that includes both partitioning and windowing.

Expected Query Result		
year	month	day
2019	8	21
2019	8	23
2019	9	12
2019	9	14
2019	9	15
2019	9	16
2019	9	17
2019	9	23
2019	9	27
2019	9	30
2019	10	1
2019	10	8
2019	10	9
2019	10	10
2019	10	28
2019	10	29
2019	10	30
2019	11	26
2019	11	28
2019	11	29
2020	1	9
2020	1	12
2020	1	13
2020	1	14
2020	3	14

Advice: Don't Plagiarize

You are encouraged to discuss solution methods with your peers, and even to look up possible solution ideas on the internet, but all of your submitted queries must be your own work. As a rule of thumb, to ensure you do not accidentally plagiarize, do not look at anyone else's queries (or allow them to see yours). Additionally, if you use a version control system (such as Github) to store your work, ensure that the repository is private. If your queries are posted in a public setting (even inadvertantly), you may become entangled in any ensuing academic integrity investigation if your work is copied by someone else.

Submission and Evaluation

This assignment will be marked through a combination of automated testing (that is, running your submitted queries and examining the result) and human inspection. To expedite the marking process (and ensure consistency between different student submissions), you are required to submit your queries for each question inside of premade query template files. Three empty template files have been posted to conneX: a4q1_queries.txt, a4q2_queries.txt and a4q3_queries.txt. Please place your query for each subquestion in the indicated spaces in the templates before submitting (we will be using an automated system to extract each query individually for marking, so failure to comply with this requirement will likely result in you receiving a mark of zero for any queries which do not meet the formatting requirements). Notice that the provided files have the .txt extension instead of .sql (which would normally be used for SQL queries); this is due to a technical limitation of conneX (which does not properly handle files with the .sql extension when they are submitted, probably because it sees them as a security risk).

You are required to submit your answers to each question in three files called a4q1_queries.txt, a4q2_queries.txt and a4q3_queries.txt, using the provided templates as a starting point. Your answer for each query must consist of a single SQL statement (which may having a WITH clause containing multiple subqueries, and/or several SELECT statements joined by set operators). As a point of reference, your answer is a 'single SQL statement' if it contains only one semicolon (at the end of the query). Although your files will be associated with your account, please ensure that each submitted file contains a comment with your name and student number.

You are permitted to delete and resubmit your submission as many times as you want before the due date, but no submissions or resubmissions will be accepted after the due date has passed. You will receive a mark of zero if you have not officially submitted your assignment (and received a confirmation email) before the due date.

Only the files that you submit through conneX will be marked. The best way to make sure your submission is correct is to download it from conneX after submitting and test it. You are not permitted to revise your submission after the due date, and late submissions will not be accepted, so you should ensure that you have submitted the correct version of your code before the due date. conneX will allow you to change your submission before the due date if you notice a mistake. After submitting your assignment, conneX will automatically send you a confirmation email. **If you do not receive such an email, you did not submit the assignment.** If you have problems with the submission process, send an email to the instructor **before** the due date.