DATA 607
Meetup 2

SQL

Andy Catlin

"Always be prepared to absorb a big hit.
Always be focused enough to create a big win.
Diversified enough to survive, concentrated enough to matter."

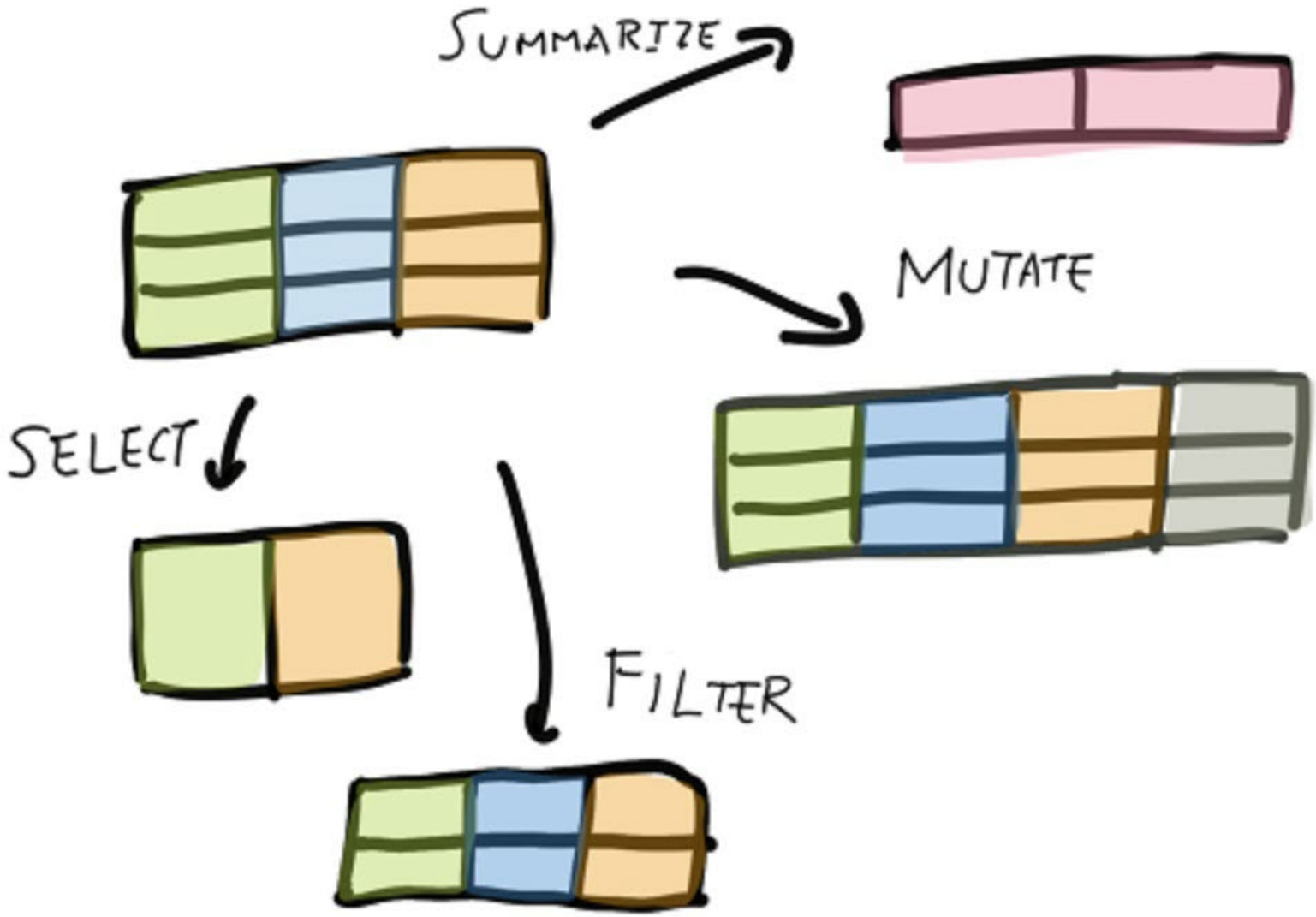
James Clear

Topics

- SQL
  - Rectangular Operations
  - SQL QUERY Logical Processing
  - Where to Perform the Work
  - Window Functions
- Data Science in Context Presentations
  - Hazal Gunduz
- Interesting Assignment 1 solutions

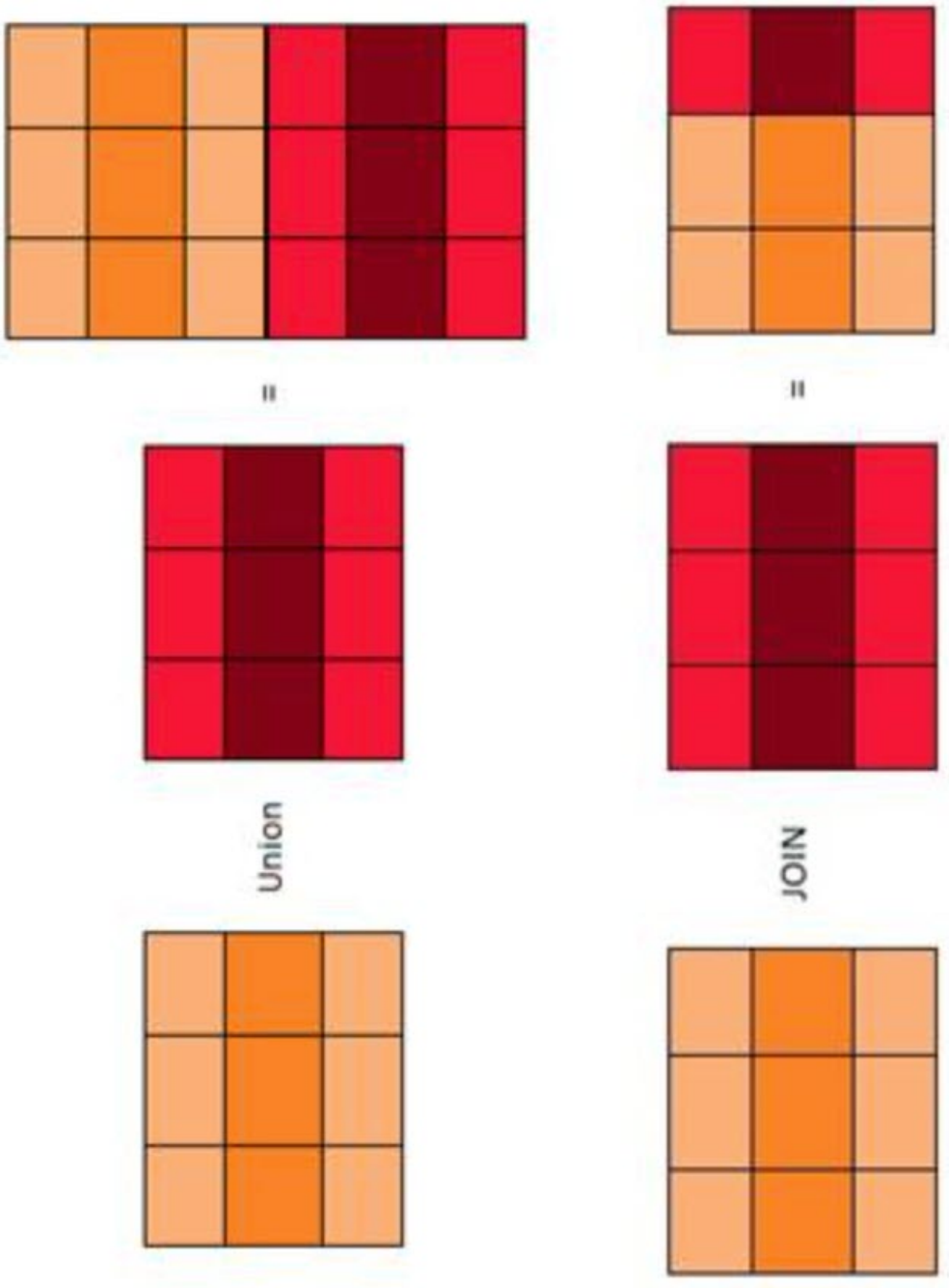
SQL

- **Rectangular Operations**
- SQL QUERY Logical Processing
- Where to Perform the Work
- Window Functions



Source: https://changhsinlee.com/pyspark-dataframe-basics/

https://www.c-sharpcorner.com/blogs/union-vs-join-in-sql



SQL

- Rectangular Operations
- **SQL QUERY Logical Processing**
- Where to Perform the Work
- Window Functions

QUERY LOGICAL PROCESSING

```
SELECT      [DISTINCT] Customer, COUNT(*) AS NumOrders
FROM        Orders
WHERE       OrderDate > '20180101'
GROUP BY    Customer
HAVING      COUNT(*) > 1
ORDER BY    NumOrders DESC
OFFSET 0 ROWS FETCH NEXT 2 ROWS ONLY
```


Orders

Order ID	Customer	Order Date
1	Dave	2018-01-01
2	John	2018-01-02
3	Gerald	2018-01-03
4	John	2018-01-09

Source: Ami Levin

In what order are the clauses of the SQL SELECT statement executed?

QUERY LOGICAL PROCESSING

```
      6
5 SELECT    [DISTINCT] Customer, COUNT(*) AS NumOrders
1 FROM      Orders
2 WHERE     OrderDate > '20180101'
3 GROUP BY  Customer
4 HAVING    COUNT(*) > 1
7 ORDER BY  NumOrders DESC
8 OFFSET 0 ROWS FETCH NEXT 2 ROWS ONLY
```

Orders

Order ID	Customer	Order Date
1	Dave	2018-01-01
2	John	2018-01-02
3	Gerald	2018-01-03
4	John	2018-01-09

Source: Ami Levin

Can I use aliases from my SELECT clause in my WHERE CLAUSE? In my ORDER BY clause?

QUERY LOGICAL PROCESSING

```
      6
5 SELECT    [DISTINCT] Customer, COUNT(*) AS NumOrders
1 FROM      Orders
2 WHERE     OrderDate > '20180101'
3 GROUP BY  Customer
4 HAVING    COUNT(*) > 1
7 ORDER BY  NumOrders DESC
8 OFFSET 0 ROWS FETCH NEXT 2 ROWS ONLY
```

Orders

Order ID	Customer	Order Date
1	Dave	2018-01-01
2	John	2018-01-02
3	Gerald	2018-01-03
4	John	2018-01-09

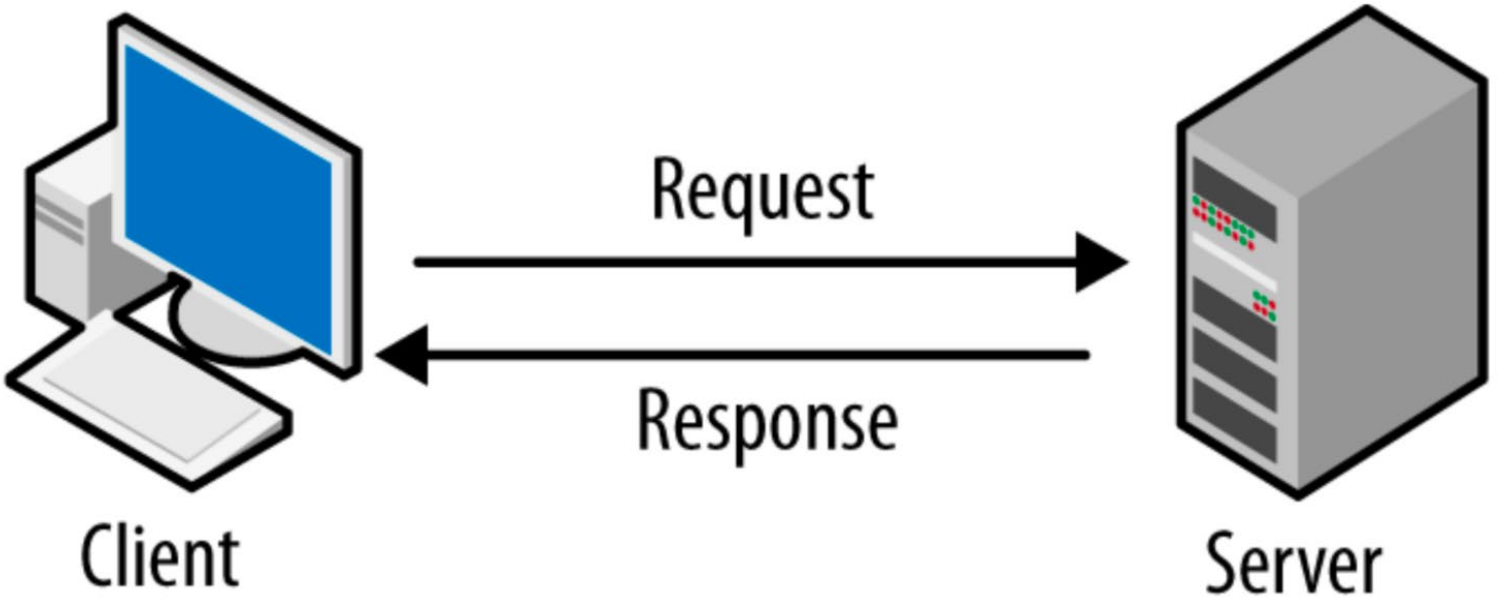
Source: Ami Levin

Can I use aggregations in my WHERE clause? In my ORDER BY clause?

SQL

- Rectangular Operations
- SQL QUERY Logical Processing
- **Where to Perform the Work**
- Window Functions

What is the work, and where should this work be performed?



Source: https://madooei.github.io/cs421_sp20_homepage/client-server-app/

SQL

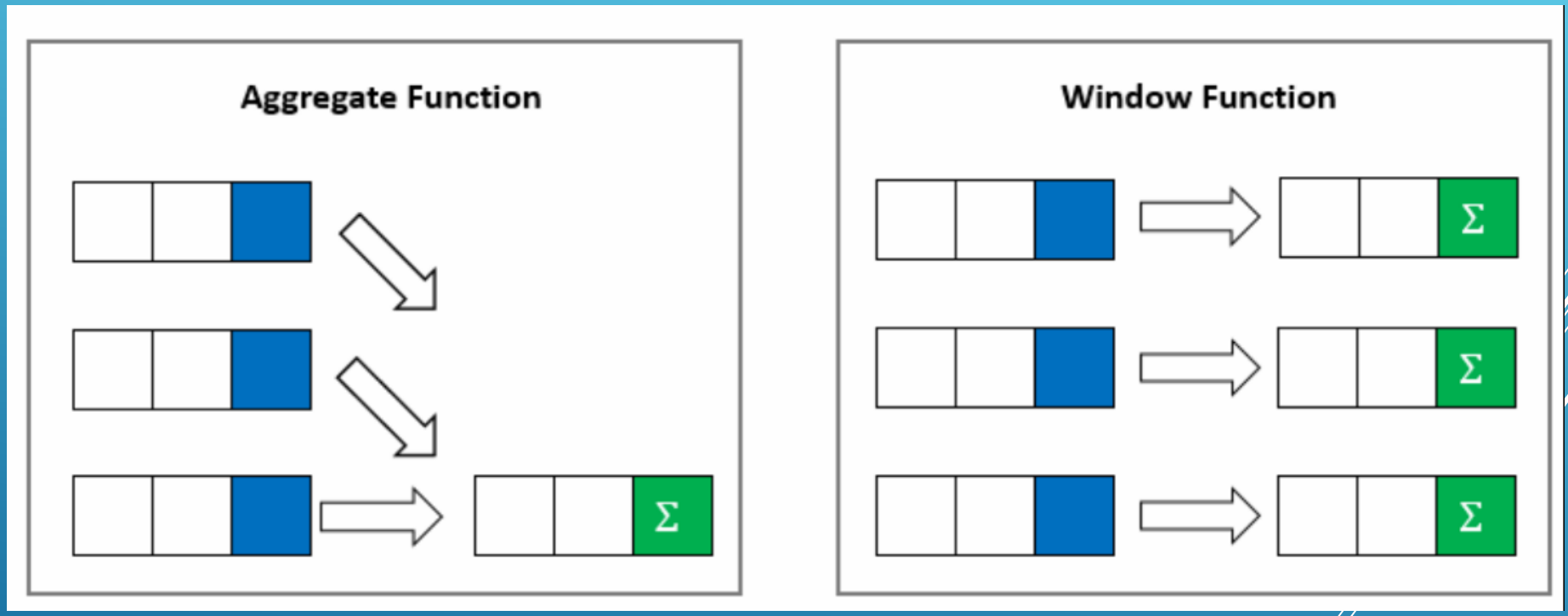
- Rectangular Operations
- SQL QUERY Logical Processing
- Where to Perform the Work
- **Window Functions**

Windows Function demonstration code

- The sample code for the windows function demonstration is in acatlin's github repo data as wf_createtables.sql and wf_samples.sql

- https://github.com/acatlin/data/blob/master/wf_createtables.sql
- https://github.com/acatlin/data/blob/master/wf_samples.sql

- 1. Learn when to use and how to write Window Functions in SQL
- 2. Apply design pattern across use cases, and across environments
  - https://dplyr.tidyverse.org/articles/window-functions.html
  - https://pandas.pydata.org/pandas-docs/stable/reference/window.html
- 3. Use test driven development and toy datasets to speed your development (and learning!) processes



https://www.sqltutorial.org/sql-window-functions/

QUERY LOGICAL PROCESSING

```
      6
5 SELECT    [DISTINCT] Customer, COUNT(*) AS NumOrders
1 FROM      Orders
2 WHERE     OrderDate > '20180101'
3 GROUP BY  Customer
4 HAVING    COUNT(*) > 1
7 ORDER BY  NumOrders DESC
8 OFFSET 0 ROWS FETCH NEXT 2 ROWS ONLY
```

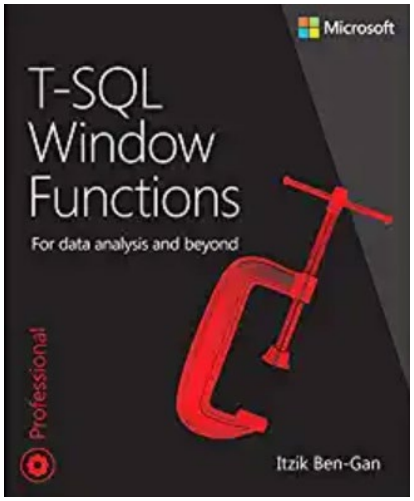
Orders

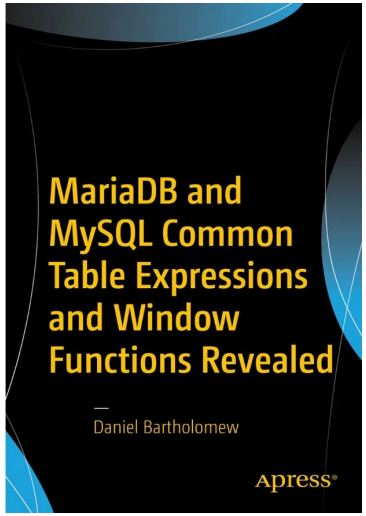
Order ID	Customer	Order Date
1	Dave	2018-01-01
2	John	2018-01-02
3	Gerald	2018-01-03
4	John	2018-01-09

Source: Ami Levin

WINDOW FUNCTIONS ARE ONLY AVAILABLE IN THE SELECT CLAUSE and after!

ANSI SQL VS. VENDOR IMPLEMENTATIONS





game_date	home_team	away_team	home_points	away_points
2019-09-29	Bills	Patriots	10	16
2019-10-20	Bills	Dolphins	31	21
2019-12-29	Bills	Jets	6	13
2019-09-15	Dolphins	Patriots	0	43
2019-11-03	Dolphins	Jets	26	18
2019-11-17	Dolphins	Bills	20	37
2019-09-08	Jets	Bills	16	17
2019-10-21	Jets	Patriots	0	33
2019-12-08	Jets	Dolphins	22	21
2019-09-22	Patriots	Jets	30	14
2019-12-21	Patriots	Bills	24	17
2019-12-29	Patriots	Dolphins	24	27

game_date	home_team	away_team	home_points	away_points	home_avg	home_differential
2019-09-29	Bills	Patriots	10	16	17.4	-7.4
2019-10-20	Bills	Dolphins	31	21	17.4	13.6
2019-12-29	Bills	Jets	6	13	17.4	-11.4
2019-09-15	Dolphins	Patriots	0	43	17.4	-17.4
2019-11-03	Dolphins	Jets	26	18	17.4	8.6
2019-11-17	Dolphins	Bills	20	37	17.4	2.6
2019-09-08	Jets	Bills	16	17	17.4	-1.4
2019-10-21	Jets	Patriots	0	33	17.4	-17.4
2019-12-08	Jets	Dolphins	22	21	17.4	4.6
2019-09-22	Patriots	Jets	30	14	17.4	12.6
2019-12-21	Patriots	Bills	24	17	17.4	6.6
2019-12-29	Patriots	Dolphins	24	27	17.4	6.6

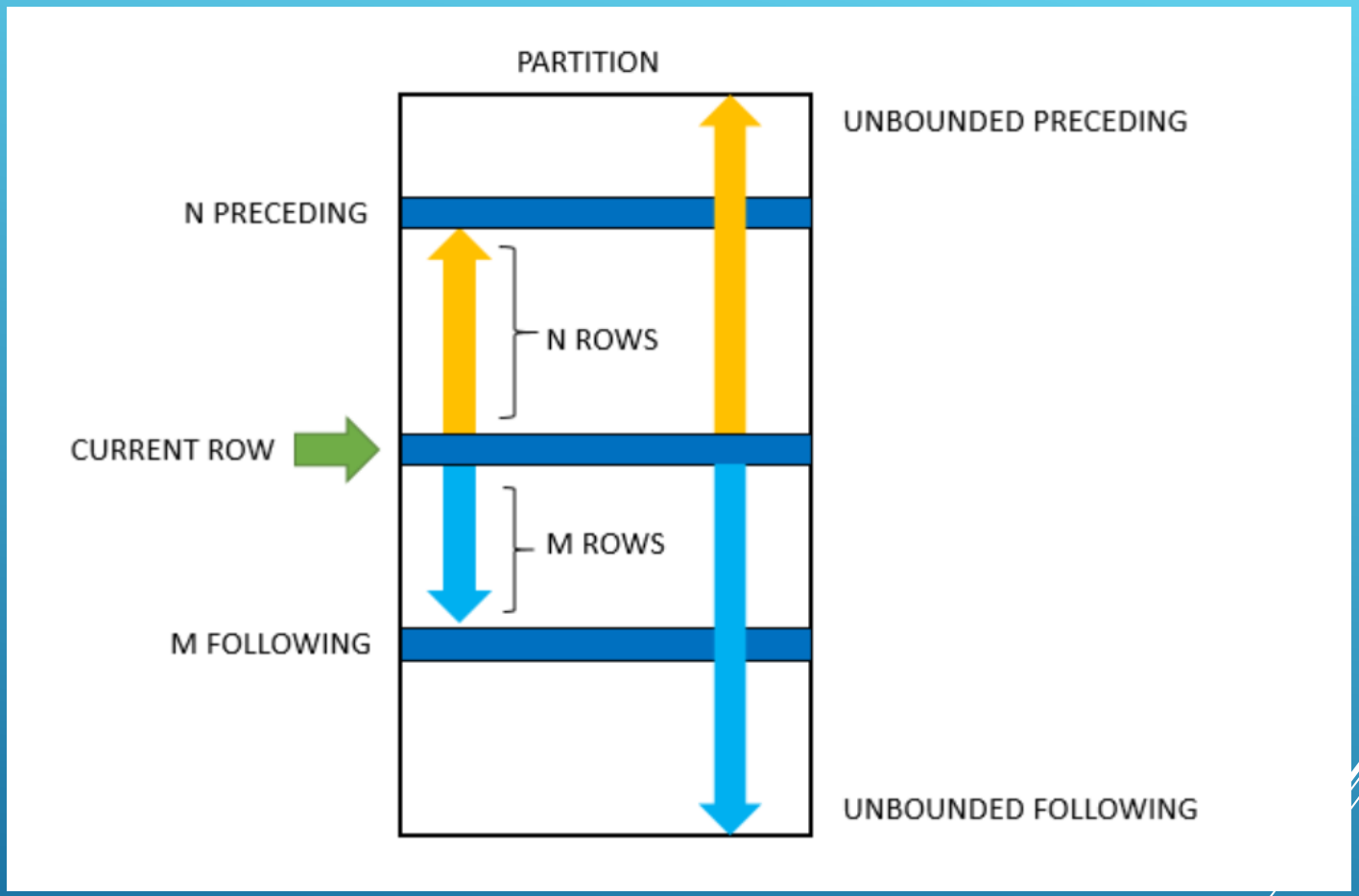
You'll write this first with a subquery; You'll then learn to write with a window function. Window functions are potentially much faster…

TASK: WRITE SQL TO DISPLAY INFORMATION ABOVE, AND TO CALCULATE NEW COLUMNS TO DISPLAY THE OVERALL AVERAGE HOME POINTS ACROSS ALL TEAMS (17.4), AND THE DIFFERENTIAL FROM EACH GAME'S SCORE TO THE OVERALL SCORE (FOR THE FIRST GAME ABOVE, -7.4)

HTTPS://DBFIDDLE.UK/

The best current SQL implementation for Window Functions is PostgreSQL

```
-- subquery in SELECT list
SELECT *,
  (SELECT round(avg(home_points),1) AS home_avg FROM
afc_east),
  home_points - (SELECT round(avg(home_points),1) FROM
afc_east) AS home_differential
FROM afc_east
  ORDER BY home_team, game_date;
```



https://www.sqltutorial.org/sql-window-functions/

game_date	home_team	away_team	home_points	away_points	cum_home_points
2019-09-29	Bills	Patriots	10	16	10
2019-10-20	Bills	Dolphins	31	21	41
2019-12-29	Bills	Jets	6	13	47
2019-09-15	Dolphins	Patriots	0	43	0
2019-11-03	Dolphins	Jets	26	18	26
2019-11-17	Dolphins	Bills	20	37	46
2019-09-08	Jets	Bills	16	17	16
2019-10-21	Jets	Patriots	0	33	16
2019-12-08	Jets	Dolphins	22	21	38
2019-09-22	Patriots	Jets	30	14	30
2019-12-21	Patriots	Bills	24	17	54
2019-12-29	Patriots	Dolphins	24	27	78

In the window function's **OVER()** clause, the window of analysis is defined by the **PARTITION BY** clause; the frame of analysis can be further limited by the **ORDER BY** and **ROWS/RANGE BETWEEN** clauses.

```
SUM(HOME_POINTS)
  OVER(PARTITION BY HOME_TEAM
    ORDER BY GAME_DATE
    ROWS BETWEEN UNBOUNDED PRECEDING
    AND CURRENT ROW)
```

game_date	home_team	away_team	home_points	away_points	cum_home_points
2019-09-29	Bills	Patriots	10	16	10
2019-10-20	Bills	Dolphins	31	21	41
2019-12-29	Bills	Jets	6	13	47
2019-09-15	Dolphins	Patriots	0	43	0
2019-11-03	Dolphins	Jets	26	18	26
2019-11-17	Dolphins	Bills	20	37	46
2019-09-08	Jets	Bills	16	17	16
2019-10-21	Jets	Patriots	0	33	16
2019-12-08	Jets	Dolphins	22	21	38
2019-09-22	Patriots	Jets	30	14	30
2019-12-21	Patriots	Bills	24	17	54
2019-12-29	Patriots	Dolphins	24	27	78

GROUP BY changes the structure of the resultset:
Many -> One

WINDOW FUNCTIONS do not change the structure of the resultset:
One -> One

```
SUM(HOME_POINTS)
  OVER(PARTITION BY HOME_TEAM
    ORDER BY GAME_DATE
    ROWS BETWEEN UNBOUNDED PRECEDING
    AND CURRENT ROW)
```

Aggregation	MIN, MAX, SUM, AVG, COUNT
Ranking	RANK, DENSE_RANK, ROW_NUMBER, NTILE
Offset	LEAD, LAG
Analytic/Distribution	PERCENT_RANK, CUME_DIST

game_date	home_team	away_team	home_points	away_points
2019-09-29	Bills	Patriots	10	16
2019-10-20	Bills	Dolphins	31	21
2019-12-29	Bills	Jets	6	13
2019-09-15	Dolphins	Patriots	0	43
2019-11-03	Dolphins	Jets	26	18
2019-11-17	Dolphins	Bills	20	37
2019-09-08	Jets	Bills	16	17
2019-10-21	Jets	Patriots	0	33
2019-12-08	Jets	Dolphins	22	21
2019-09-22	Patriots	Jets	30	14
2019-12-21	Patriots	Bills	24	17
2019-12-29	Patriots	Dolphins	24	27

game_date	home_team	away_team	home_points	away_points
2019-09-29	Bills	Patriots	10	16
2019-10-20	Bills	Dolphins	31	21
2019-12-29	Bills	Jets	6	13
2019-09-15	Dolphins	Patriots	0	43
2019-11-03	Dolphins	Jets	26	18
2019-11-17	Dolphins	Bills	20	37
2019-09-08	Jets	Bills	16	17
2019-10-21	Jets	Patriots	0	33
2019-12-08	Jets	Dolphins	22	21
2019-09-22	Patriots	Jets	30	14
2019-12-21	Patriots	Bills	24	17
2019-12-29	Patriots	Dolphins	24	27

TASK: USING A WINDOW FUNCTION, WRITE SQL TO DISPLAY INFORMATION ABOVE, AND TO CALCULATE NEW COLUMNS TO DISPLAY THE OVERALL AVERAGE HOME POINTS ACROSS ALL TEAMS(17.4), AND THE DIFFERENTIAL FROM EACH GAME'S SCORE TO THE OVERALL SCORE (FOR THE FIRST GAME ABOVE, -7.4)

game_date	home_team	away_team	home_points	away_points	home_avg	home_differential
2019-09-29	Bills	Patriots	10	16	17.4	-7.4
2019-10-20	Bills	Dolphins	31	21	17.4	13.6
2019-12-29	Bills	Jets	6	13	17.4	-11.4
2019-09-15	Dolphins	Patriots	0	43	17.4	-17.4
2019-11-03	Dolphins	Jets	26	18	17.4	8.6
2019-11-17	Dolphins	Bills	20	37	17.4	2.6
2019-09-08	Jets	Bills	16	17	17.4	-1.4
2019-10-21	Jets	Patriots	0	33	17.4	-17.4
2019-12-08	Jets	Dolphins	22	21	17.4	4.6
2019-09-22	Patriots	Jets	30	14	17.4	12.6
2019-12-21	Patriots	Bills	24	17	17.4	6.6
2019-12-29	Patriots	Dolphins	24	27	17.4	6.6

WHAT IF WE WANT TO CALCULATE (OVERALL *OR FOR EACH TEAM*) THE DIFFERENTIAL IN HOME AVERAGE POINTS SCORED, OR THE RANK OF HOME POINTS SCORED, OR THE CUMULATIVE HOME POINTS SCORED SEASON-TO-DATE, OR…

```
-- SUBQUERY IN SELECT list
SELECT *,
  (SELECT round(avg(home_points),1) AS home_avg FROM
afc_east),
  home_points - (SELECT round(avg(home_points),1) FROM
afc_east) AS home_differential
FROM afc_east
  ORDER BY home_team, game_date;
```

```
-- window function
SELECT *,
  ROUND(AVG(home_points)
      OVER(),1)
      AS home_avg,
  ROUND(home_points -
     AVG(home_points) OVER(),1)
     AS home_differential
FROM afc_east
  ORDER BY home_team, game_date;
```

game_date	home_team	away_team	home_points	away_points	home_avg	home_differential
2019-09-29	Bills	Patriots	10	16	15.7	-5.7
2019-10-20	Bills	Dolphins	31	21	15.7	15.3
2019-12-29	Bills	Jets	6	13	15.7	-9.7
2019-09-15	Dolphins	Patriots	0	43	15.3	-15.3
2019-11-03	Dolphins	Jets	26	18	15.3	10.7
2019-11-17	Dolphins	Bills	20	37	15.3	4.7
2019-09-08	Jets	Bills	16	17	12.7	3.3
2019-10-21	Jets	Patriots	0	33	12.7	-12.7
2019-12-08	Jets	Dolphins	22	21	12.7	9.3
2019-09-22	Patriots	Jets	30	14	26.0	4.0
2019-12-21	Patriots	Bills	24	17	26.0	-2.0
2019-12-29	Patriots	Dolphins	24	27	26.0	-2.0

DESCRIBE IN WORDS: WHAT IS HOME_DIFFERENTIAL?

game_date	home_team	away_team	home_points	away_points	home_avg	home_differential
2019-09-29	Bills	Patriots	10	16	15.7	-5.7
2019-10-20	Bills	Dolphins	31	21	15.7	15.3
2019-12-29	Bills	Jets	6	13	15.7	-9.7
2019-09-15	Dolphins	Patriots	0	43	15.3	-15.3
2019-11-03	Dolphins	Jets	26	18	15.3	10.7
2019-11-17	Dolphins	Bills	20	37	15.3	4.7
2019-09-08	Jets	Bills	16	17	12.7	3.3
2019-10-21	Jets	Patriots	0	33	12.7	-12.7
2019-12-08	Jets	Dolphins	22	21	12.7	9.3
2019-09-22	Patriots	Jets	30	14	26.0	4.0
2019-12-21	Patriots	Bills	24	17	26.0	-2.0
2019-12-29	Patriots	Dolphins	24	27	26.0	-2.0

AVG(HOME_POINTS) OVER(PARTITION BY HOME_TEAM)

game_date	home_team	away_team	home_points	away_points	home_rank
2019-09-29	Bills	Patriots	10	16	9
2019-10-20	Bills	Dolphins	31	21	1
2019-12-29	Bills	Jets	6	13	10
2019-09-15	Dolphins	Patriots	0	43	11
2019-11-03	Dolphins	Jets	26	18	3
2019-11-17	Dolphins	Bills	20	37	7
2019-09-08	Jets	Bills	16	17	8
2019-10-21	Jets	Patriots	0	33	11
2019-12-08	Jets	Dolphins	22	21	6
2019-09-22	Patriots	Jets	30	14	2
2019-12-21	Patriots	Bills	24	17	4
2019-12-29	Patriots	Dolphins	24	27	4

DESCRIBE IN WORDS: WHAT IS HOME_RANK?

game_date	home_team	away_team	home_points	away_points	home_rank
2019-09-29	Bills	Patriots	10	16	9
2019-10-20	Bills	Dolphins	31	21	1
2019-12-29	Bills	Jets	6	13	10
2019-09-15	Dolphins	Patriots	0	43	11
2019-11-03	Dolphins	Jets	26	18	3
2019-11-17	Dolphins	Bills	20	37	7
2019-09-08	Jets	Bills	16	17	8
2019-10-21	Jets	Patriots	0	33	11
2019-12-08	Jets	Dolphins	22	21	6
2019-09-22	Patriots	Jets	30	14	2
2019-12-21	Patriots	Bills	24	17	4
2019-12-29	Patriots	Dolphins	24	27	4

RANK()
OVER(ORDER BY HOME_POINTS DESC)

game_date	home_team	away_team	home_points	away_points	home_rank
2019-09-29	Bills	Patriots	10	16	2
2019-10-20	Bills	Dolphins	31	21	1
2019-12-29	Bills	Jets	6	13	3
2019-09-15	Dolphins	Patriots	0	43	3
2019-11-03	Dolphins	Jets	26	18	1
2019-11-17	Dolphins	Bills	20	37	2
2019-09-08	Jets	Bills	16	17	2
2019-10-21	Jets	Patriots	0	33	3
2019-12-08	Jets	Dolphins	22	21	1
2019-09-22	Patriots	Jets	30	14	1
2019-12-21	Patriots	Bills	24	17	2
2019-12-29	Patriots	Dolphins	24	27	2

DESCRIBE IN WORDS: WHAT IS HOME_RANK?

game_date	home_team	away_team	home_points	away_points	home_rank
2019-09-29	Bills	Patriots	10	16	2
2019-10-20	Bills	Dolphins	31	21	1
2019-12-29	Bills	Jets	6	13	3
2019-09-15	Dolphins	Patriots	0	43	3
2019-11-03	Dolphins	Jets	26	18	1
2019-11-17	Dolphins	Bills	20	37	2
2019-09-08	Jets	Bills	16	17	2
2019-10-21	Jets	Patriots	0	33	3
2019-12-08	Jets	Dolphins	22	21	1
2019-09-22	Patriots	Jets	30	14	1
2019-12-21	Patriots	Bills	24	17	2
2019-12-29	Patriots	Dolphins	24	27	2

```
RANK()
  OVER(PARTITION BY HOME_TEAM
  ORDER BY HOME_POINTS DESC)
```

game_date	home_team	away_team	home_points	away_points	cum_home_points
2019-09-29	Bills	Patriots	10	16	10
2019-10-20	Bills	Dolphins	31	21	41
2019-12-29	Bills	Jets	6	13	47
2019-09-15	Dolphins	Patriots	0	43	0
2019-11-03	Dolphins	Jets	26	18	26
2019-11-17	Dolphins	Bills	20	37	46
2019-09-08	Jets	Bills	16	17	16
2019-10-21	Jets	Patriots	0	33	16
2019-12-08	Jets	Dolphins	22	21	38
2019-09-22	Patriots	Jets	30	14	30
2019-12-21	Patriots	Bills	24	17	54
2019-12-29	Patriots	Dolphins	24	27	78

DESCRIBE IN WORDS: WHAT IS CUM_HOME_POINTS?

game_date	home_team	away_team	home_points	away_points	cum_home_points
2019-09-29	Bills	Patriots	10	16	10
2019-10-20	Bills	Dolphins	31	21	41
2019-12-29	Bills	Jets	6	13	47
2019-09-15	Dolphins	Patriots	0	43	0
2019-11-03	Dolphins	Jets	26	18	26
2019-11-17	Dolphins	Bills	20	37	46
2019-09-08	Jets	Bills	16	17	16
2019-10-21	Jets	Patriots	0	33	16
2019-12-08	Jets	Dolphins	22	21	38
2019-09-22	Patriots	Jets	30	14	30
2019-12-21	Patriots	Bills	24	17	54
2019-12-29	Patriots	Dolphins	24	27	78

```
SUM(HOME_POINTS)
  OVER(PARTITION BY HOME_TEAM
    ORDER BY GAME_DATE
    ROWS BETWEEN UNBOUNDED PRECEDING
    AND CURRENT ROW)
```

WHAT ARE SOME OTHER INTERESTING USE CASES FOR WINDOW FUNCTIONS?

OPTIONAL EXTRA CREDIT ASSIGNMENT (10 POINTS / 1%)

- 1. Find a dataset that includes a time series (e.g. end of day stock prices for several instruments).
- 2. Use window functions to calculate year to date and six day moving averages.
- 3. Present your code in our next meetup in a three to five minute presentation.
- 4. You may work in a small team on this assignment. ☺

Some Exemplary Assignment 1 solutions

- Cassandra Boylan, https://rpubs.com/r_amazonian/DATA-607_Assignment-1 High risk COVID
- Shariq Mian, https://rpubs.com/mianshariq/Shariq_Mian_Data_607_Assignment_1 Baseball ELO data; exemplary question-driven
- Jeffrey Parks, https://rpubs.com/jefedigital/covid-tracker-538 COVID data; business-level (code-suppressed)
- Peter Phung, https://rpubs.com/peterphung2043/803170 COVID polls. Kable,boxplots in grid
- Santiago Torres, https://rpubs.com/sserrot/803175 Marriage isn't dead yet