TAP TO GO BACK TO KIOSK MENU

SAS® GLOBAL FORUM 2020

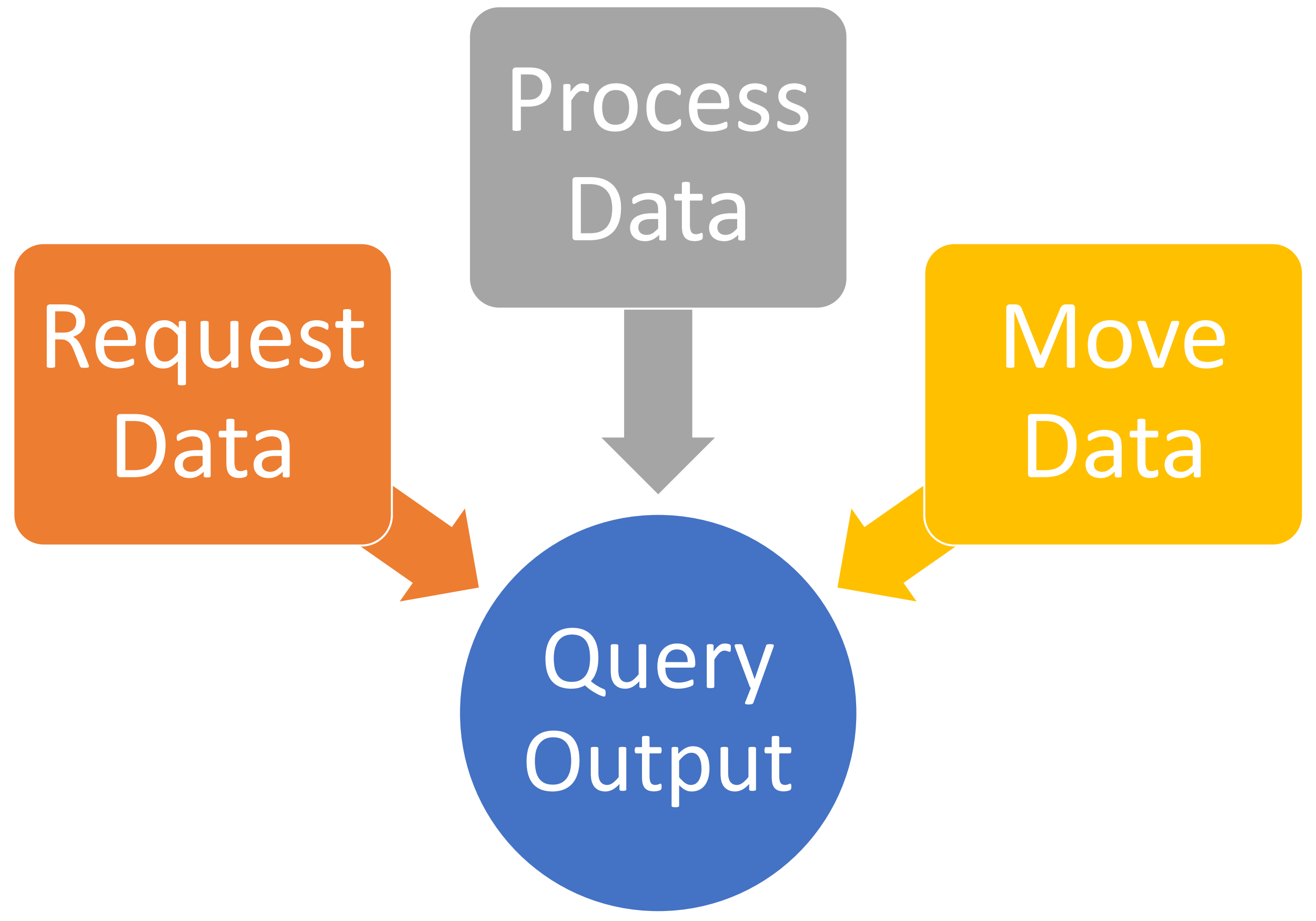
MARCH 29 – APRIL 1
WASHINGTON, DC

USERS PROGRAM

Why is it taking so long!? Utilizing query optimization to pull from external sources

TAP TO GO BACK TO KIOSK MENU

Kurt Stultz
IDEXX Laboratories



Kurt Stultz

Pulling data from external sources is a challenging job for any data analyst, but writing queries to do it quickly and consistently often separates the wheat from the chaff. As a company that extracts data from various, disparate data sources, our team took on the responsibility to learn the underpinning of querying in order to optimize our data requests.

After developing a foundational understanding of querying (mainly in SQL), we identified two main areas that often take up the majority of the querying processing time: 1) data transportation and 2) SQL processing. We discovered that if we were able to simplify the amount of data being transported and being processed, our query speeds would improve. We utilized three methods of query optimization to speed up our queries

1) **Utilizing passthrough queries:** Rather than pulling a large volume of data into SAS to join or process, SAS allows us to pass a query into external data sources to fully utilize their partitioning/indexing, and minimize the amount of data that must be transported to SAS.

2) **Optimizing SQL statements:** Simply joining two large tables together can be computationally expensive, but queries can be written in a way to minimize the processing that must be done.

3) **Partitioning queries:** Chunking query requests can help avoid any memory limits that could be hindering the processing of large data pulls. If you can trim your query requests to pull data in bite-sized portions, you can merge data once it reaches SAS.

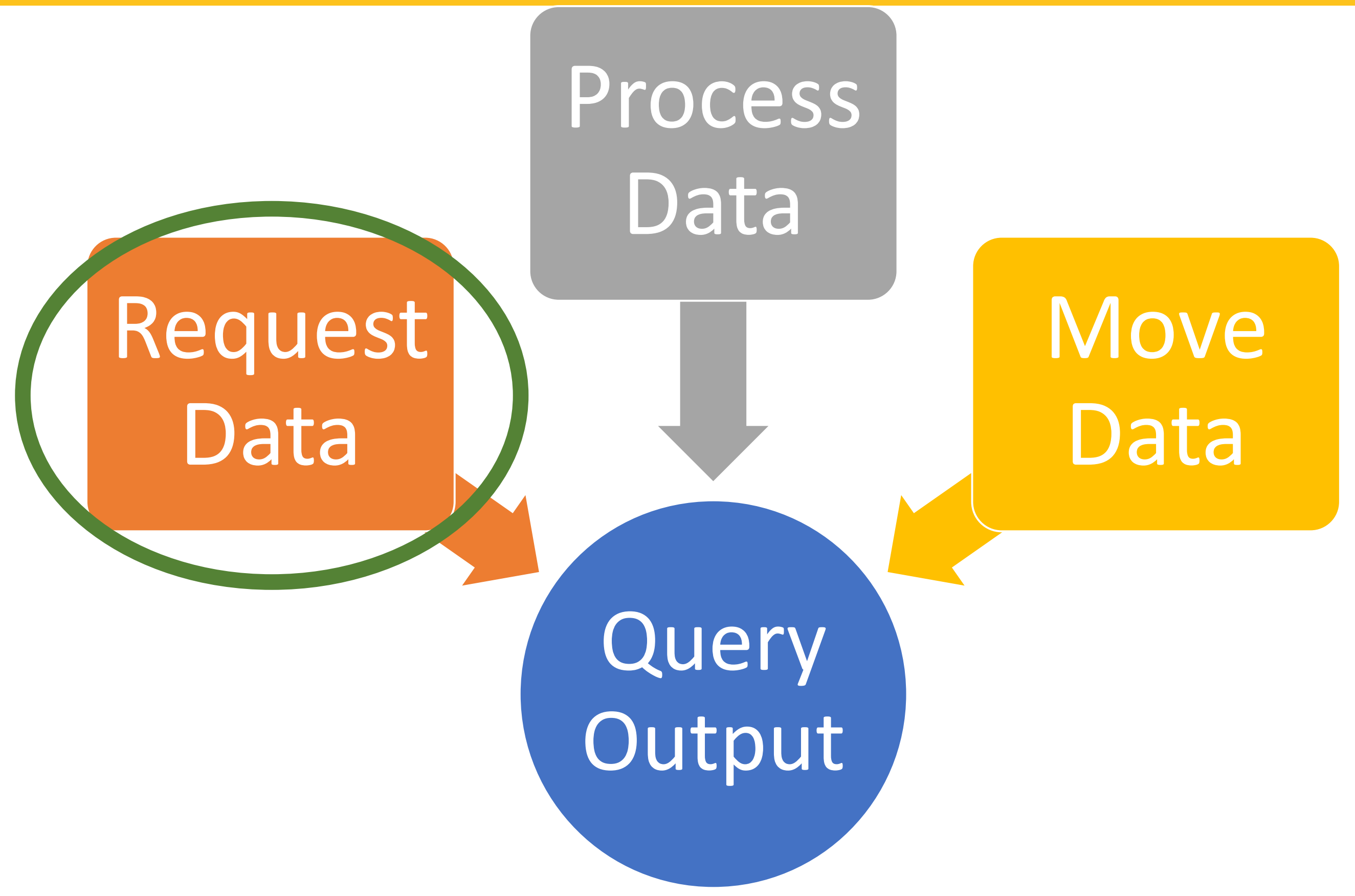
By utilizing the three methods above, we improved our query speed by anywhere from 2x to 50x. Once we understood how queries are processed, we were able to write queries that were quicker and less memory-intensive.

Who are we:
IDEXX Laboratories, Inc. is an American multinational corporation engaged in the development, manufacture, and distribution of products and services for the companion animal veterinary, livestock and poultry, water testing, and dairy markets.

Why is it taking so long!? Utilizing query optimization to pull from external sources

Kurt Stultz
IDEXX Laboratories

- Abstract
- **Request Data**
- Process Data
- Move Data
- Conclusion

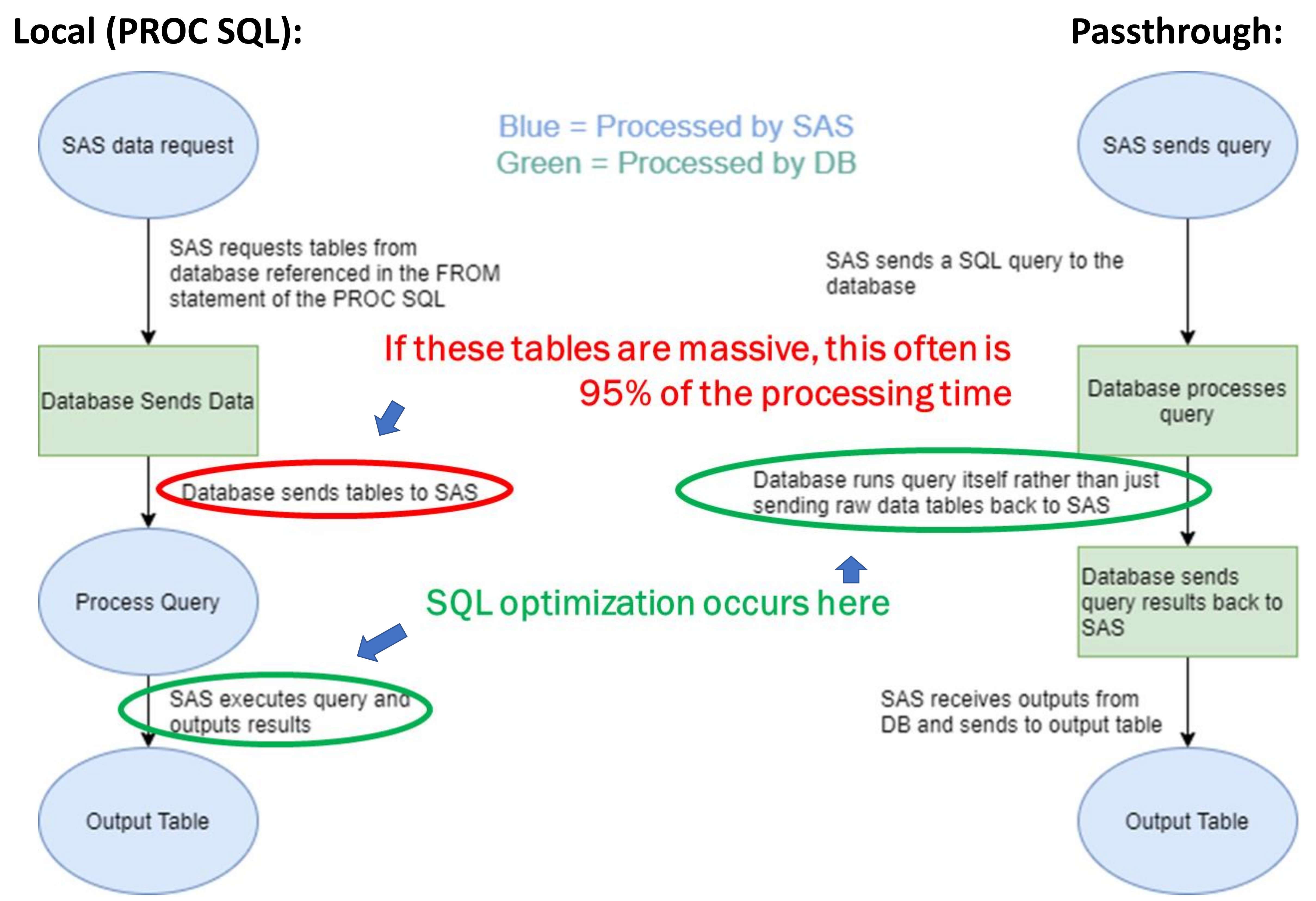


What is a passthrough query? A passthrough query changes the location of execution of a query and consequentially, the order of operations of a data request. Instead of running your query locally in SAS, a passthrough sends the SQL syntax directly to the server/database hosting the data. This allows for processing/aggregating to be done before the data is transferred to SAS.

What is different about a passthrough query

```
PROC SQL;
Create table test as
   SELECT ….
;
Quit;
```

Minimal improvement through SQL optimization


Local (PROC SQL):
Passthrough:
Blue = Processed by SAS
Green = Processed by DB
SAS data request
SAS requests tables from database referenced in the FROM statement of the PROC SQL
Database Sends Data
Database sends tables to SAS
Process Query
SAS executes query and outputs results
Output Table
If these tables are massive, this often is 95% of the processing time
SQL optimization occurs here
SAS sends query
SAS sends a SQL query to the database
Database processes query
Database runs query itself rather than just sending raw data tables back to SAS
Database sends query results back to SAS
SAS receives outputs from DB and sends to output table
Output Table

```
PROC SQL;
connect to oracle (…);
select * from connection
to oracle(
   SELECT ….
 );
disconnect from oracle;
Quit;
```

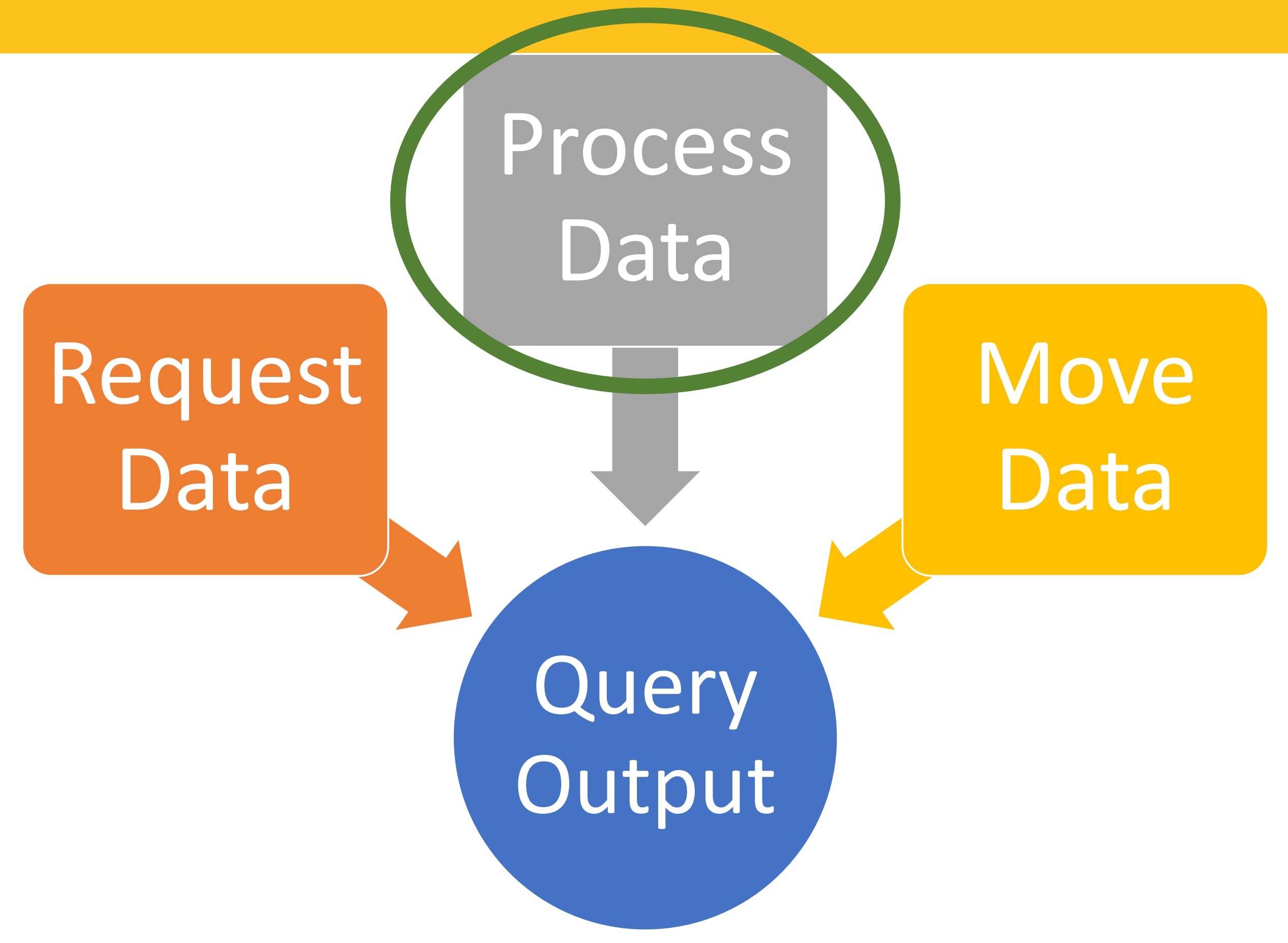
improvement through SQL optimization

Often **10x** faster

** Side note: Since the query will be processed on another server, the SQL must match the language of the server. Many things are the exact same, but there are few differences depending on the database*

Why is it taking so long!? Utilizing query optimization to pull from external sources

Kurt Stultz
IDEXX Laboratories



Processing order of a SQL statement

ORDER	CLAUSE	FUNCTION
1	from	Choose and join tables to get base data.
2	where	Filters the base data.
3	group by	Aggregates the base data.
4	having	Filters the aggregated data.
5	select	Returns the final data.
6	order by	Sorts the final data.
7	limit	Limits the returned data to a row count.

SQL Statement:

```
(5) Select a.a, a.b, count(b.c)
(1) From Table1 a    inner join Table2 b on
a.primaryKey = b.foreignKey
(2) Where a.b = 'Alpha'
(3) Group by a.a, a.b
(4) Having a.a = 'Beta'
(6) Order By a.b
(7) Limit 20
```

Not Optimized:

Join data

Filter

```
Select a.DateA, b.*

From Table1 a inner join Table2 b on
a.primaryKey = b.foreignKey

Where a.dateA > '01DEC2019'
And b.dateB > '20DEC2019'
And a.field = 'Alpha'
And b.field = 'Beta'
;
```

Optimized:

Filter data

Join

```
Select a.DateA, b.DateB, b.field
From (
     Select PrimaryKey, DateA
     From Table1
     Where
       dateA > '01DEC2019'
       And field = 'Alpha') a
  inner join on (
     Select foreignKey, DateB, field
     From Table1
     Where
       dateB > '20DEC2019'
       And field = 'Beta') b
  on a.primaryKey = b.foreignKey
;
```

A SQL statement processes in a specific order, so understanding the order can direct our optimization because we can understand what data exists in each step

```
Select a.DateA, a.Alpha
From Table1 a
Where a.beta in (
  Select distinct fieldB
  From table2
)
Quit;
```

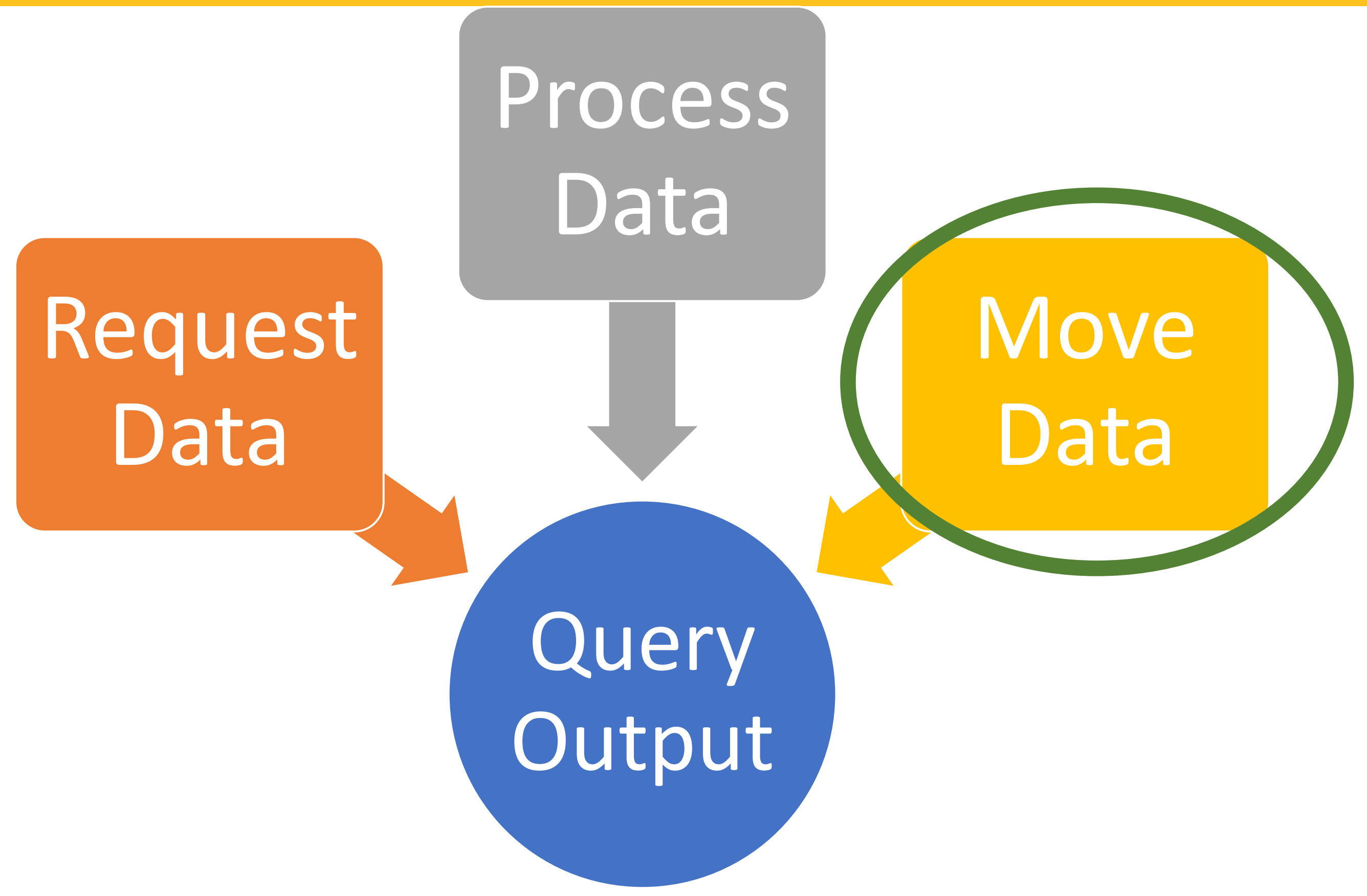
Tips for query optimization:

- Use pass-throughs: Often SQL optimization is mostly washed out if you do a local SAS query when pulling from a large data source.
- Start with the FROM statement: this is the first step to be processed and limits the volume of data pulled into the rest of your query.
- Use * sparingly: though it is more wordy, this often pulls in the maximum amount of data that exists and can easily slow your query down.
- Be aware of field length: often fields sizes are not set appropriately for columns. check or potentially set the length manually with a "cast" statement.
- Test pulling raw vs aggregated data: often we need to aggregate data at some point. Depending on the situation, the query might perform better if you pull and aggregate in the same query, but it might work better to pull raw data and *then* aggregate when you have the raw data in SAS.
- Where in (Select…): often if you need to filter by a large list of values, you can do a subquery in a where statement which avoids costly joins and allows you to filter by an extended list.

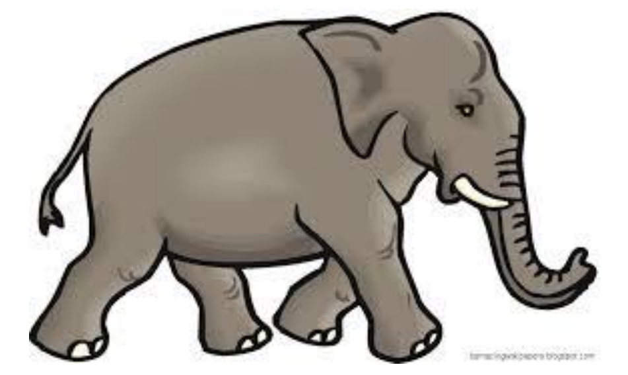
Why is it taking so long!? Utilizing query optimization to pull from external sources

Kurt Stultz
IDEXX Laboratories

- Abstract
- Request Data
- Process Data
- **Move Data**
- Conclusion



Often if you are trying to return too much data or doing too much calculating, you can run out of memory, and the query will fail.
Consider chunking your query



VS

Non Chunked:

Pull all data at once

Can often fail because it is pulling too much data for your server/session to handle

```
Select …
From…
Where
 date Between
  ’01JAN2019’ and
  ’31DEC2019’
```

Chunked:

Loop for Every Month between start and end date:

More code, but uses a fraction of the memory because we are only pulling a fraction of the data at a time

```
Do i= 1 to 12:
Create table loop_&i. as
 Select …
 From…
 Where
  date Between
   <Start of Month i> and
   <End of Month i>
End

Append all created tables
```

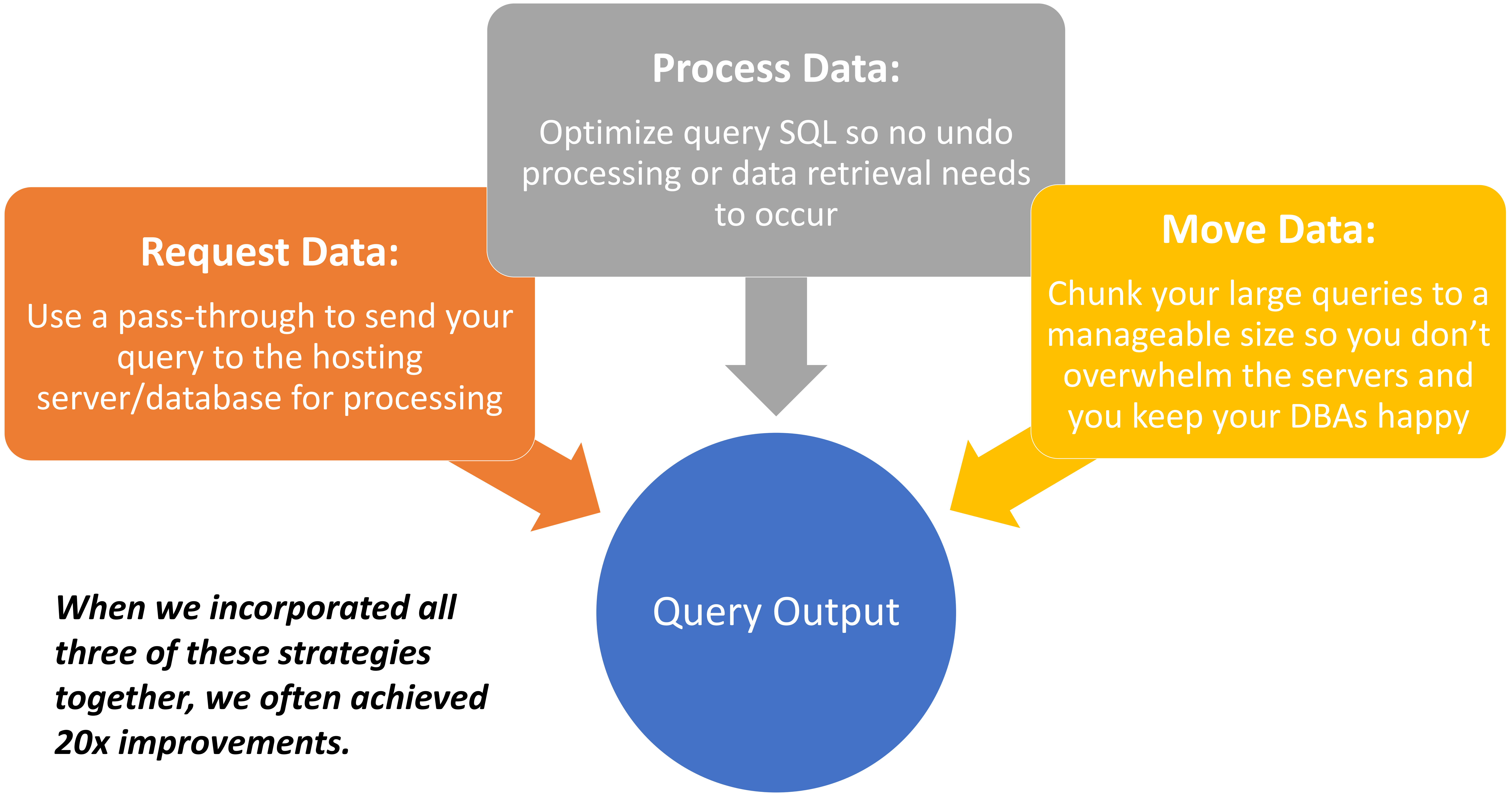
Why not create a macro?

```
%macro queryChunck(start,end,TP, tableName);
  %let loop = %SYSFUNC(intck(&TP,&start.,&end.));
  %let dayOff = %SYSFUNC(ifn(&TP = Day,2,1));
  %DO from=1 %TO &loop.+1;
    %let sDate = %sysfunc(intnx(&TP.,&end.,1-&from.,B),MMDDYY10.);
    %let eDate = %sysfunc(intnx(&TP.,&end.,&dayOff - &from.,E),MMDDYY10.);

    PROC SQL;
      connect to oracle
      create table Work.&tableName._Loop_&sDate. As
      select * from connection to oracle
      (
        SELECT *
        FROM db.table
        WHERE added_date between &sDate. and &eDate.
      disconnect from oracle;
    QUIT;
  %END;

  Data &tableName;
    set &tableName._Loop:;
  run;

proc datasets lib=work nolist;
    delete &tableName._Loop:;
  run;
%mend queryChunck;
%queryChunck('20AUG2019'd, '26AUG2019'd, Day, chunkOutput);
```

Why is it taking so long!? Utilizing query optimization to pull from external sources

Kurt Stultz
IDEXX Laboratories

Putting it all together



Process Data:

Optimize query SQL so no undo processing or data retrieval needs to occur

Request Data:

Use a pass-through to send your query to the hosting server/database for processing

Move Data:

Chunk your large queries to a manageable size so you don't overwhelm the servers and you keep your DBAs happy

When we incorporated all three of these strategies together, we often achieved 20x improvements.

SAS® GLOBAL FORUM 2020

USERS PROGRAM

MARCH 29 – APRIL 1 | WASHINGTON, DC | #SASGF