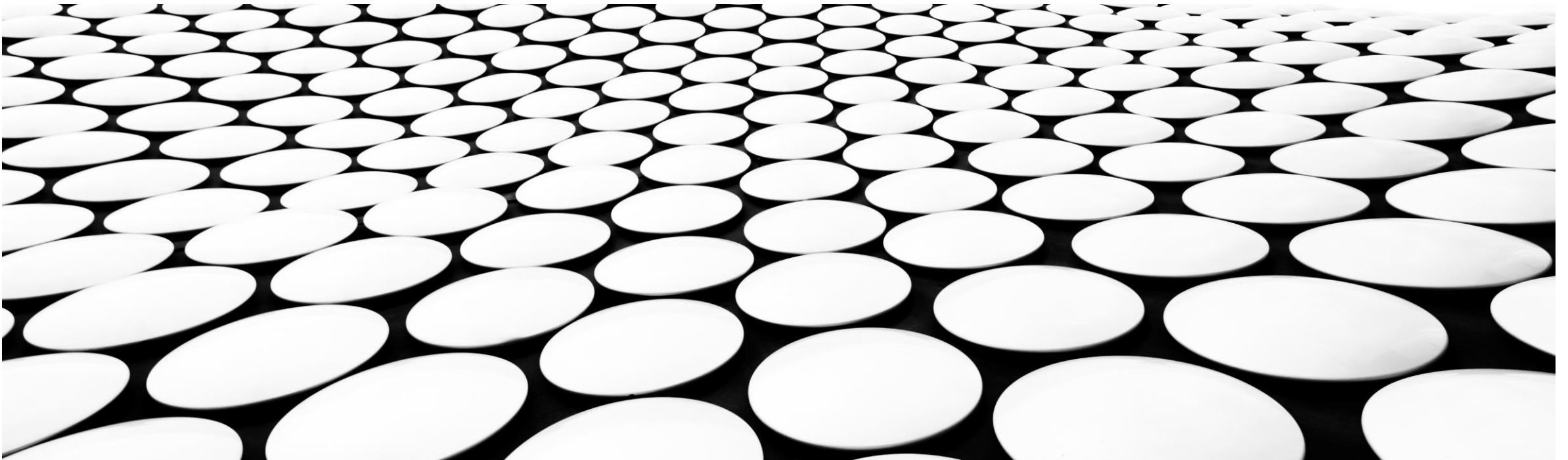
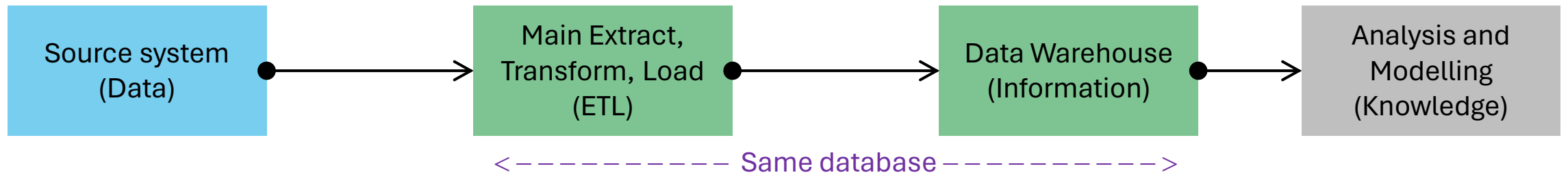

IMPACT OF SQL QUERY DESIGN AND INDEX USAGE ON DATA EXTRACTION DURATION

JOHN WONG, 2 MAY 2024

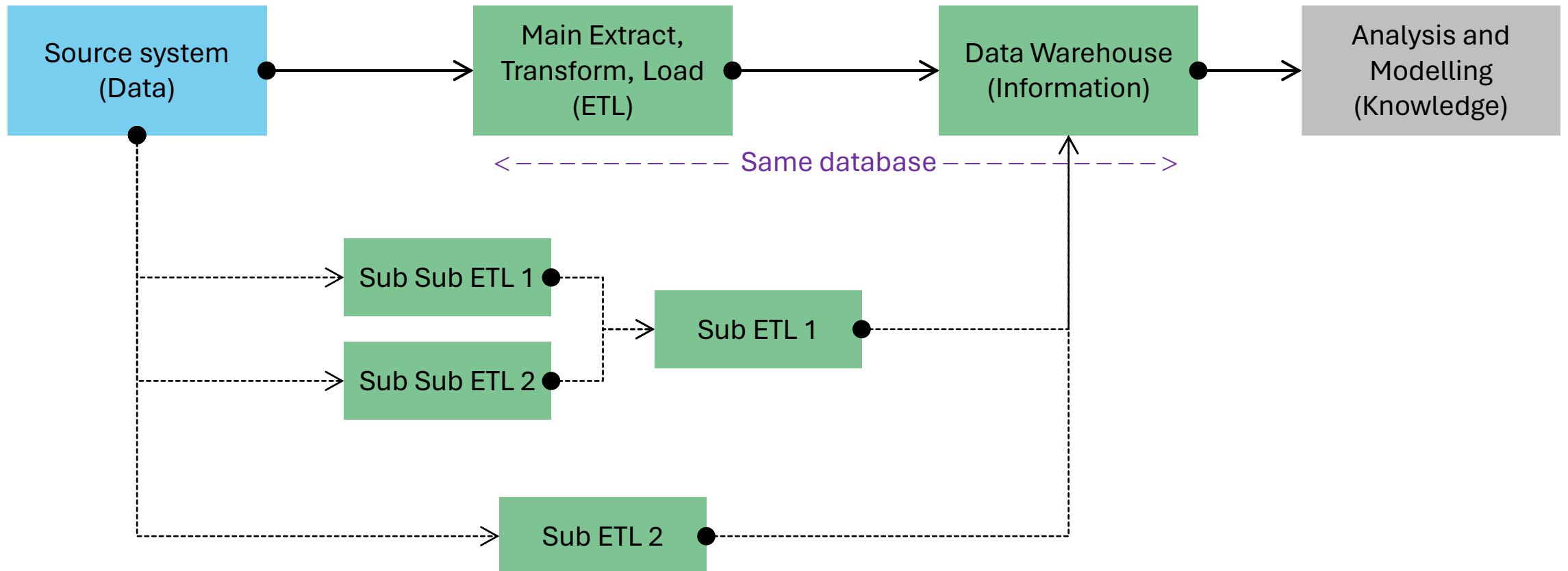


BACKGROUND

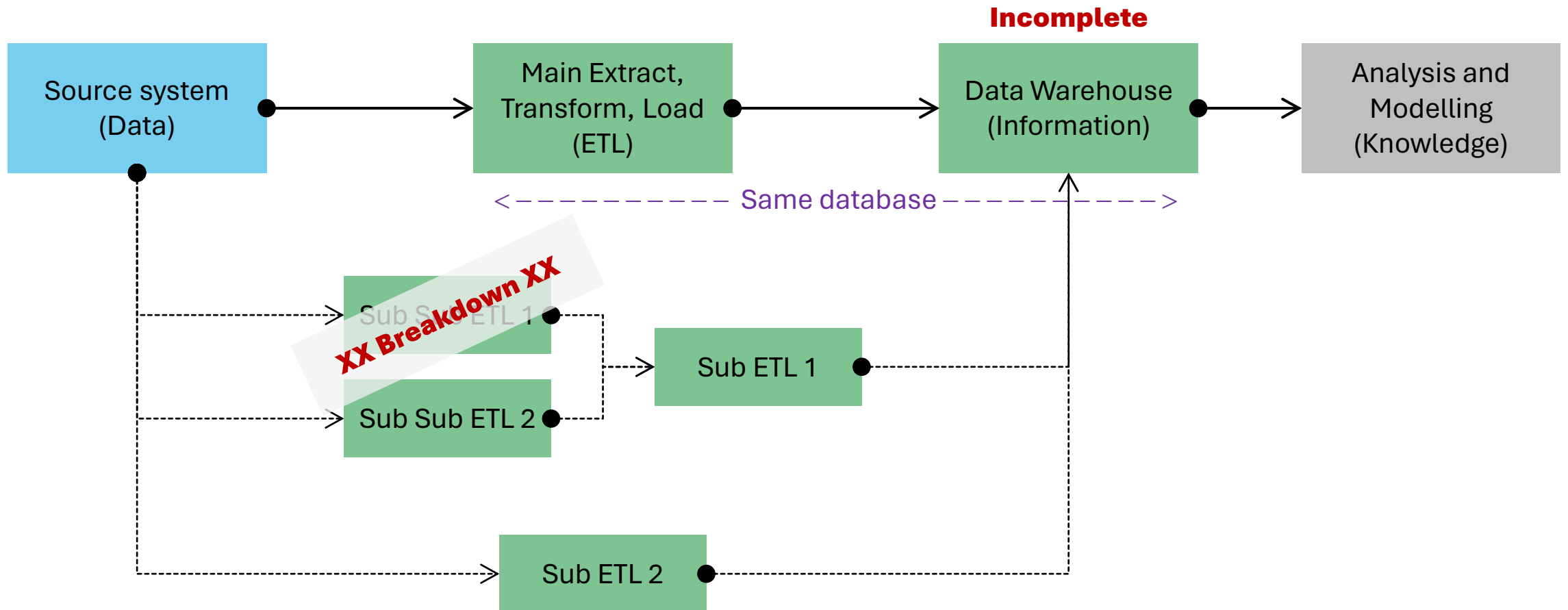
ETL PROCESS – DATA > INFORMATION > KNOWLEDGE



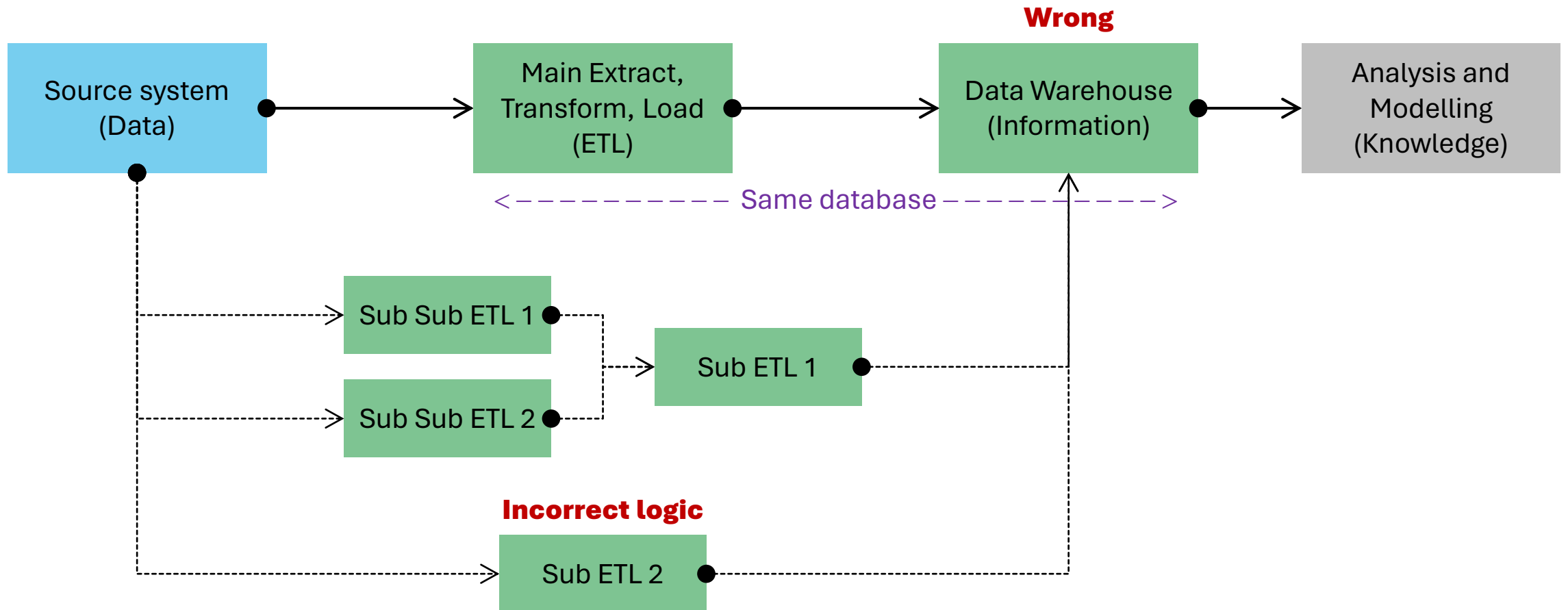
ETL PROCESS – “MAIN” CALLS WORKFLOWS “SUB” AND “SUB SUB”



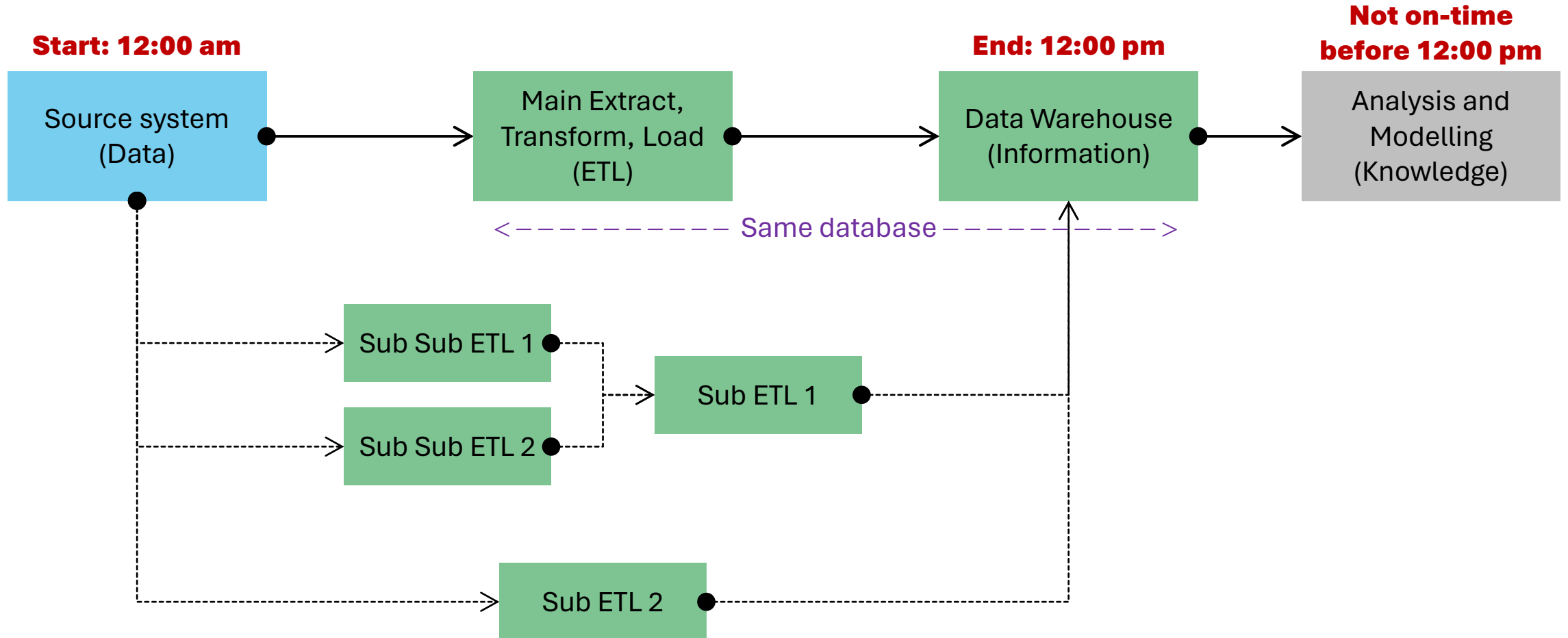
ETL PROCESS – WHAT CAN GO WRONG



ETL PROCESS – WHAT CAN GO WRONG



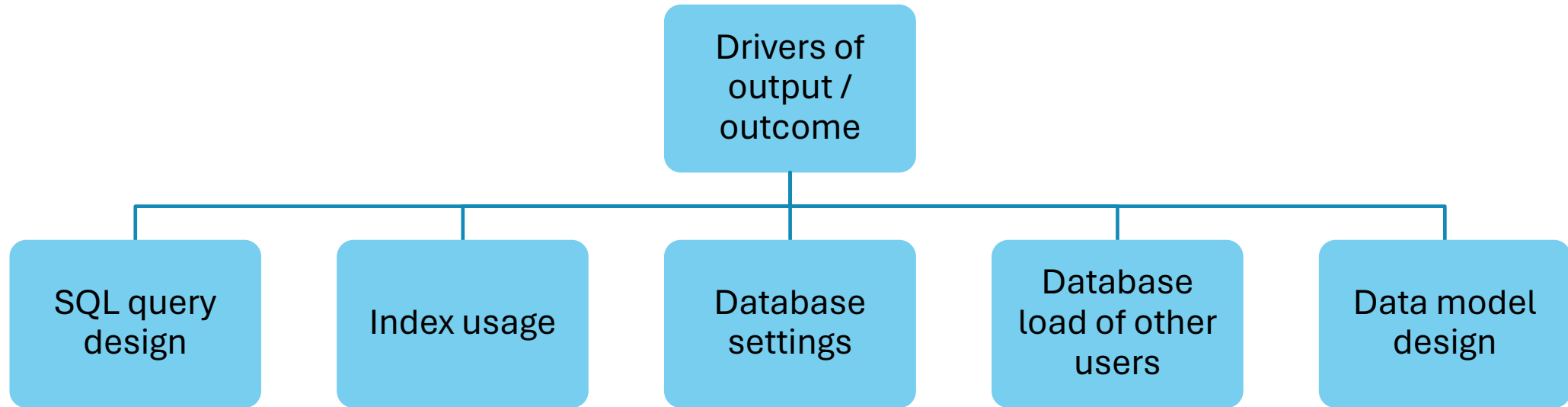
ETL PROCESS – WHAT CAN GO WRONG



ETL PROCESS – DESIRED OUTPUT / OUTCOME

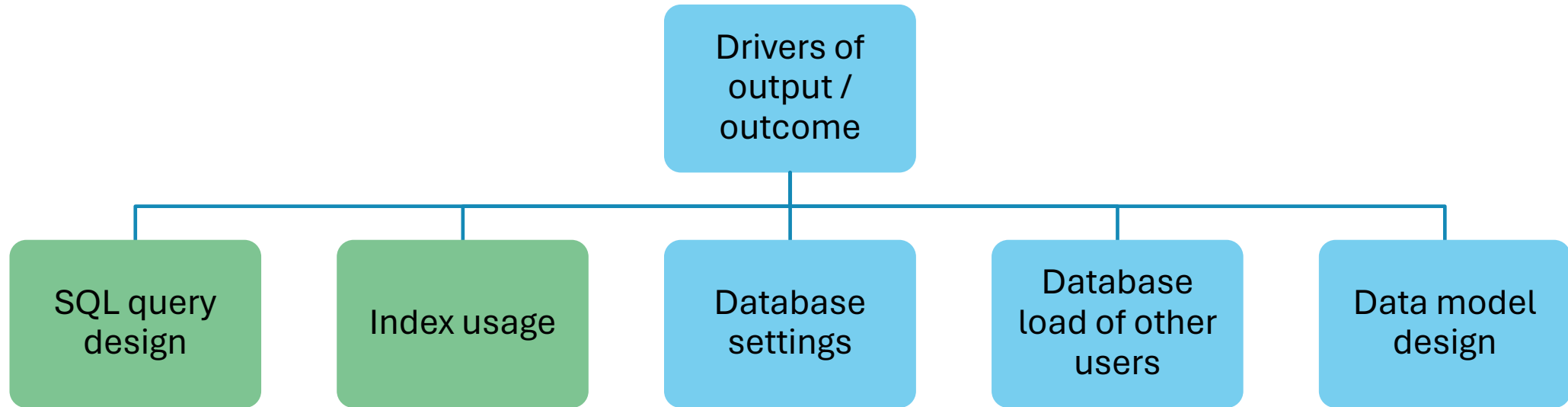
- Information is complete
- Information is correct
- Information is on-time
- Information acquisition is low-cost (move, compute, store)
- Information acquisition is low-load on database for other users

ETL PROCESS – DRIVERS OF DESIRED OUTPUT / OUTCOME



OBJECTIVE

EXPLORE IMPACT OF SQL QUERY DESIGN AND INDEX USAGE ON DATA EXTRACTION DURATION



Design of experiment (DOE) to explore

LEFT JOIN versus INNER JOIN

WHERE versus ON

Common Table Expression (CTE) versus Sub-Query

Automatic index versus manual index versus no index

Create Table versus memory

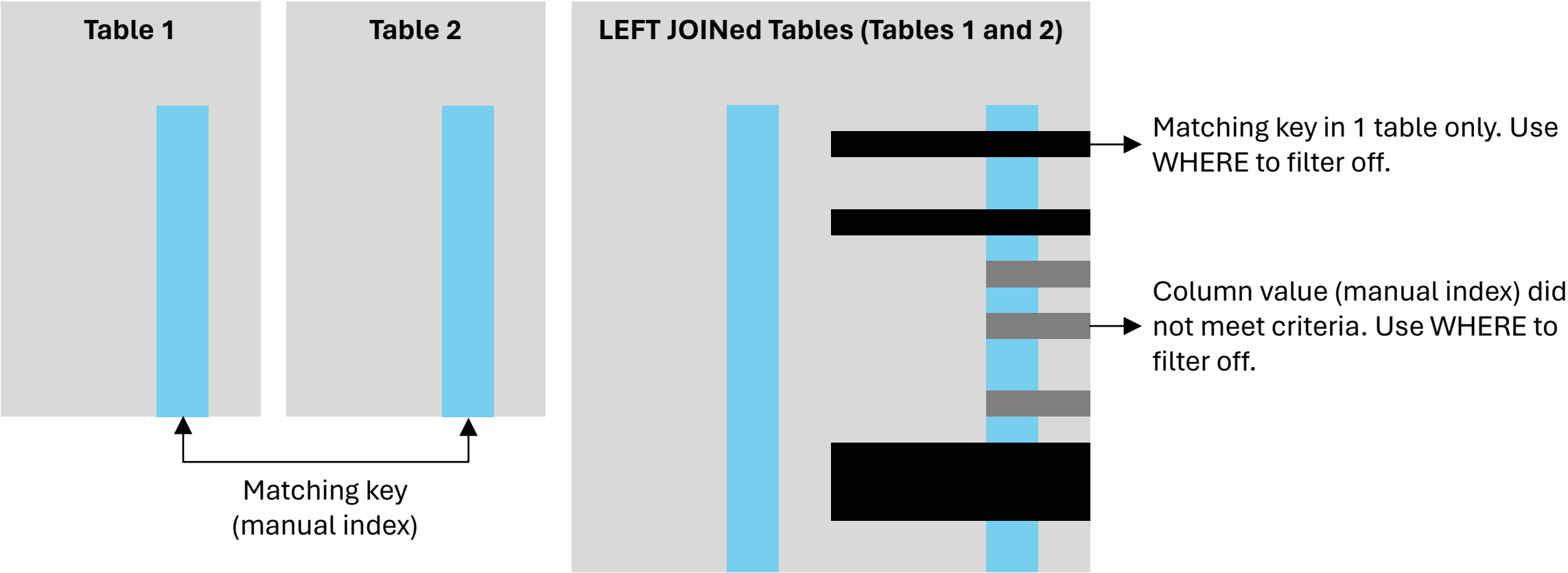
METHOD

DOE 1 METHOD – INDEX USAGE

- Join 2 tables using SQL query design with LEFT JOIN and WHERE
- Retain rows of LEFT JOINed tables where 2 tables have same matching key
- Retain rows of LEFT JOINed tables where column value meets criteria
- Create manual index on matching keys of 2 tables and column value of 1 table

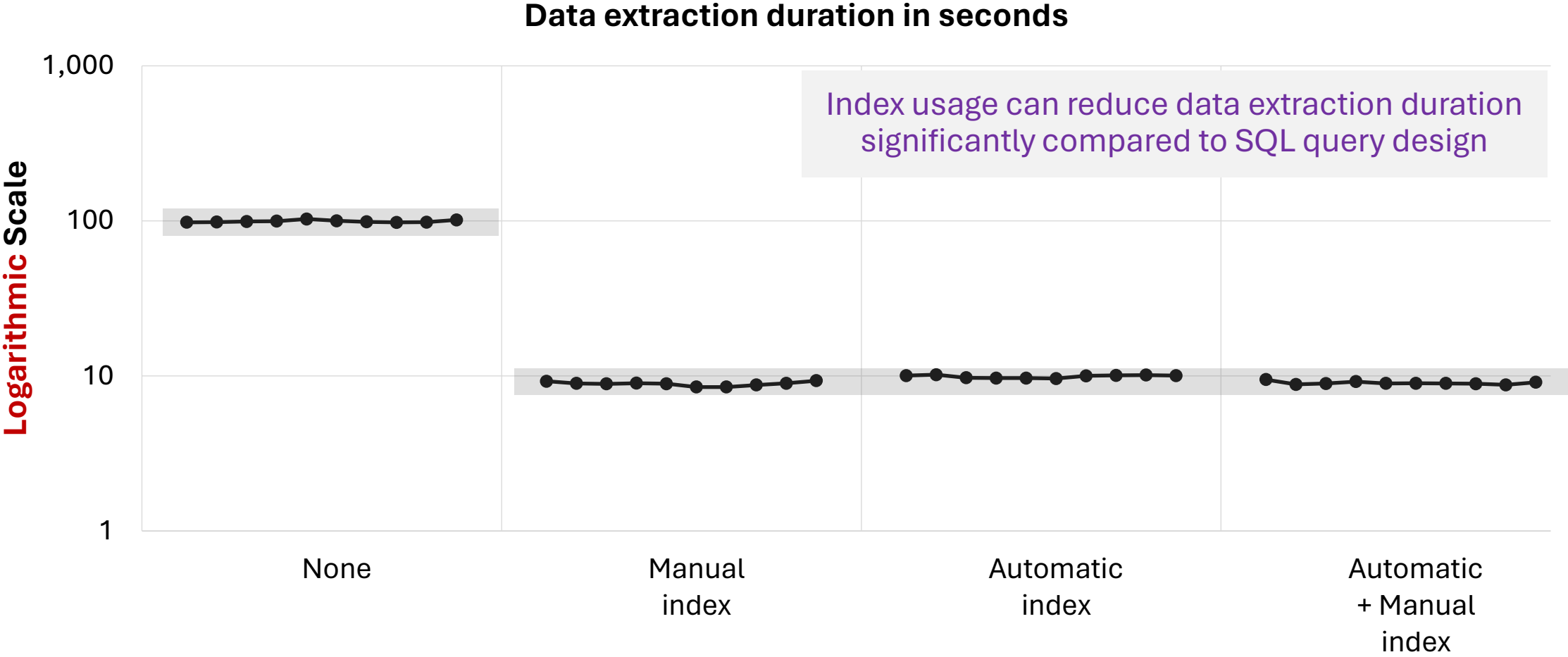
Run number	Automatic index	Manual index	Data extraction duration
Open database > 1 to 10 > Close database	No	No	
Open database > 1 to 10 > Close database	No	Yes	
Open database > 1 to 10 > Close database	Yes	No	
Open database > 1 to 10 > Close database	Yes	Yes	

DOE 1 METHOD – INDEX USAGE

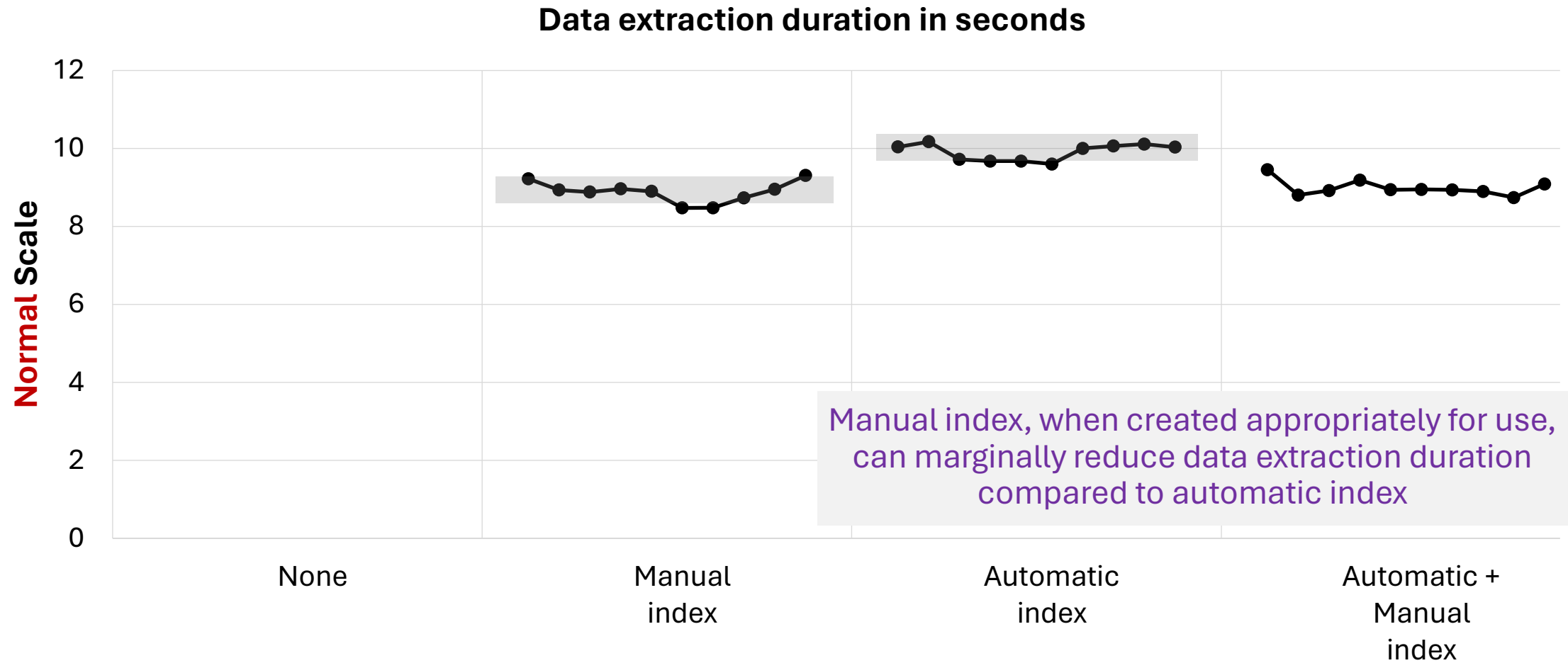


RESULT

DOE 1 RESULT – INDEX USAGE



DOE 1 RESULT – INDEX USAGE



METHOD

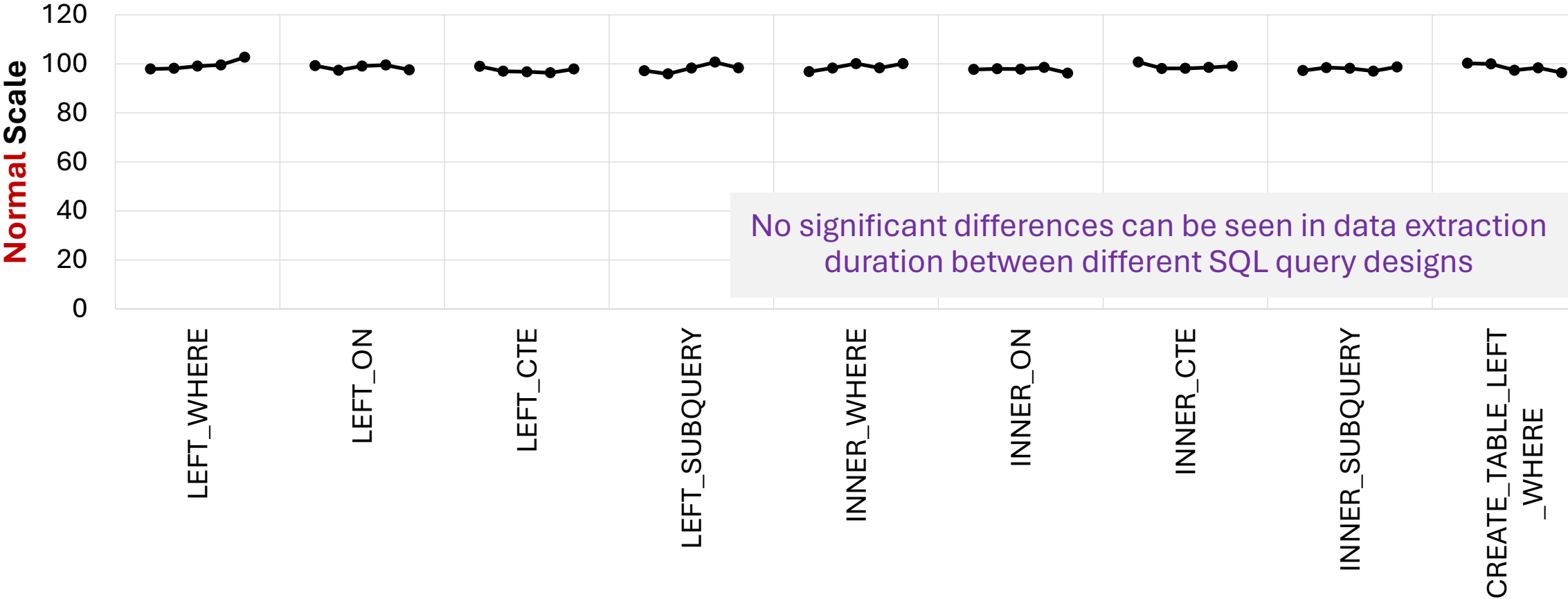
DOE 2 METHOD – SQL QUERY DESIGN

SQL query design	Run number	Automatic index	Manual index	Data extraction duration
LEFT JOIN, WHERE	Open database > 1 to 5 > Close database	No	No	
LEFT JOIN, ON	Open database > 1 to 5 > Close database	No	No	
LEFT JOIN, CTE	Open database > 1 to 5 > Close database	No	No	
LEFT JOIN, SUBQUERY	Open database > 1 to 5 > Close database	No	No	
INNER JOIN, WHERE	Open database > 1 to 5 > Close database	No	No	
INNER JOIN, ON	Open database > 1 to 5 > Close database	No	No	
INNER JOIN, CTE	Open database > 1 to 5 > Close database	No	No	
INNER JOIN, SUBQUERY	Open database > 1 to 5 > Close database	No	No	
CREATE TABLE, LEFT JOIN, WHERE	Open database > 1 to 5 > Close database	No	No	

RESULT

DOE 2 RESULT – SQL QUERY DESIGN

Data extraction duration in seconds



CONCLUSION

CONCLUSION

- Index usage can reduce data extraction duration significantly compared to SQL query design
- When manual index created appropriately for use, it can reduce data extraction duration marginally compared to automatic index
- When data extraction duration is short, it can also reduce load on database for other users

DB BROWSER FOR SQLITE

MANUAL INDEX

File Edit View Tools Help

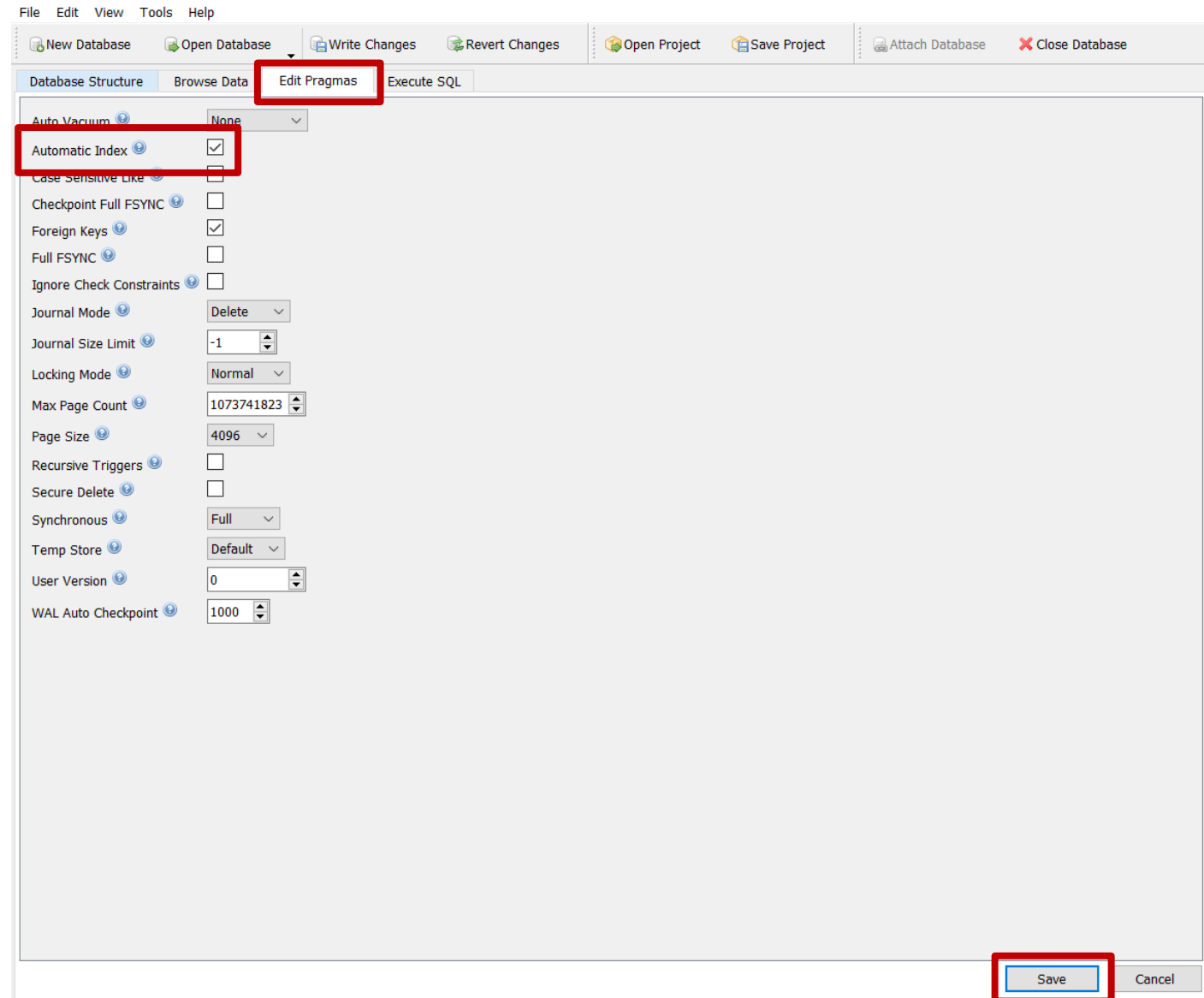
New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragmas Execute SQL

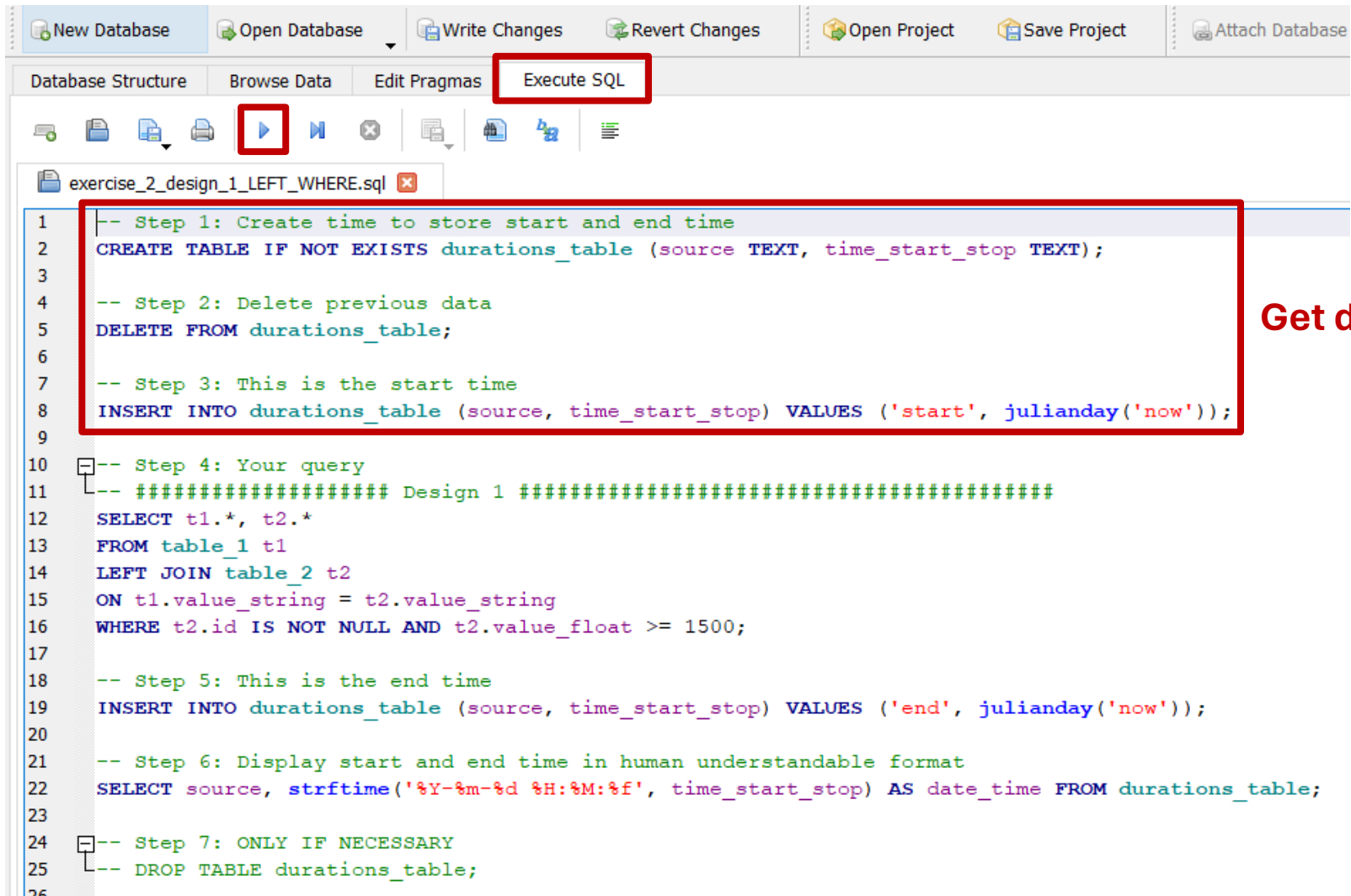
Create Table Create Index Print

Name	Type	Schema
▼ Tables (4)		
> durations_table		CREATE TABLE durations_table (source TEXT, time_start_stop TEXT)
> table_1		CREATE TABLE "table_1" ("id" INTEGER, "value_integer" INTEGER, "value_string" TEXT, PRIMARY KEY("id"))
> table_2		CREATE TABLE "table_2" ("id" INTEGER, "value_integer" INTEGER, "value_string" TEXT, "value_float" REAL, PRIMARY KEY("id"))
> table_3		CREATE TABLE table_3(id INT, value_integer INT, value_string TEXT, value_float REAL)
▼ Indices (3)		
> idx_t1_string		CREATE INDEX "idx_t1_string" ON "table_1" ("value_string" ASC)
> idx_t2_float		CREATE INDEX "idx_t2_float" ON "table_2" ("value_float" ASC)
> idx_t2_string		CREATE INDEX "idx_t2_string" ON "table_2" ("value_string" ASC)
Views (0)		
Triggers (0)		

AUTOMATIC INDEX



SQL QUERY WITH DATA EXTRACTION DURATION

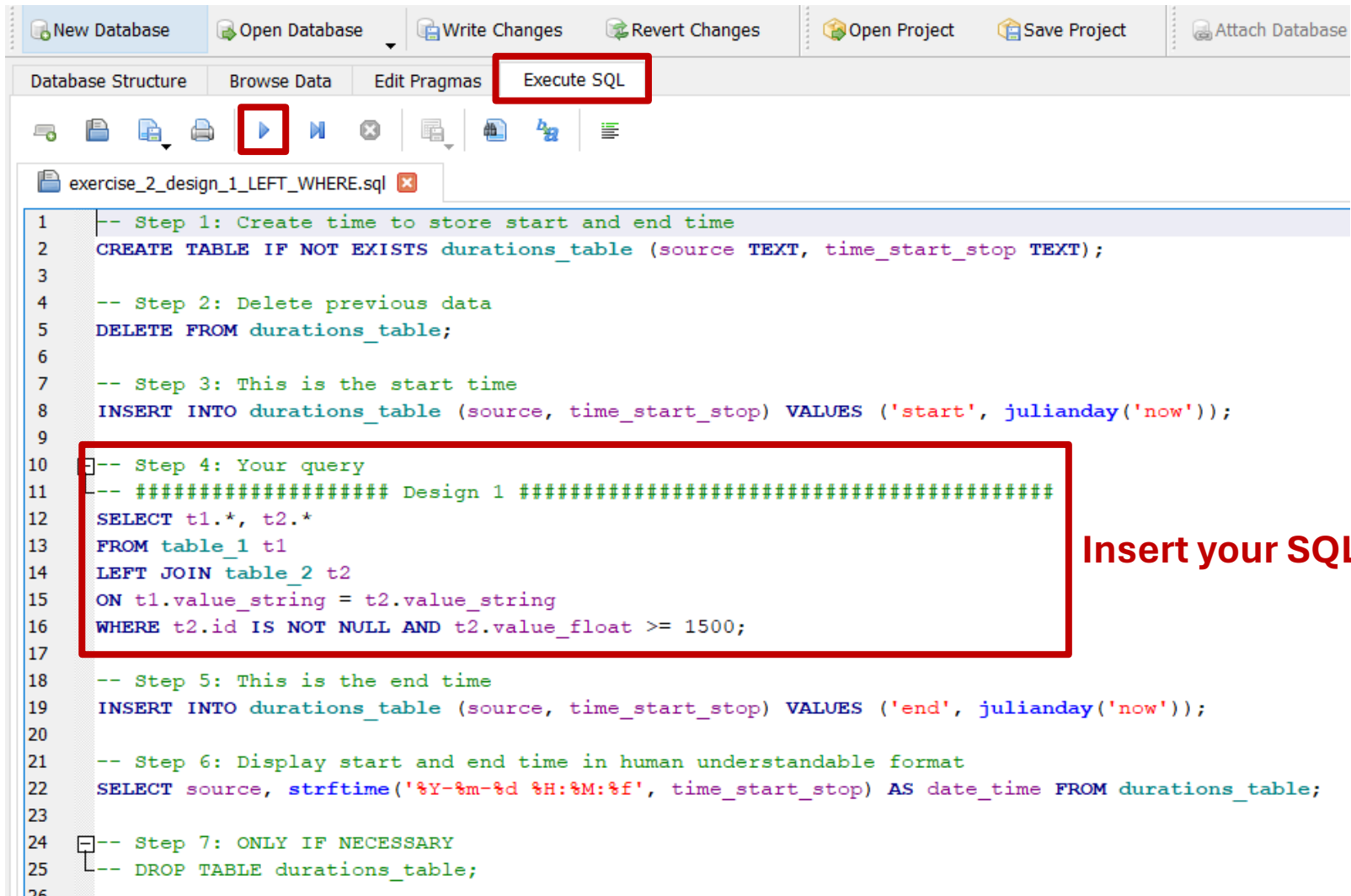


The screenshot shows a database management tool interface. At the top, there is a menu bar with options: New Database, Open Database, Write Changes, Revert Changes, Open Project, Save Project, and Attach Database. Below the menu bar is a toolbar with icons for various database operations. The 'Execute SQL' button is highlighted with a red box. Below the toolbar is a text area containing an SQL query. The query is divided into several steps, each starting with a comment. The first step is to create a table named 'durations_table' with columns 'source' and 'time_start_stop'. The second step is to delete previous data. The third step is to insert the start time into the 'durations_table' using the 'julianday('now')' function. The fourth step is to run a query that selects data from 'table_1' and 'table_2' based on certain conditions. The fifth step is to insert the end time into the 'durations_table' using the 'julianday('now')' function. The sixth step is to display the start and end times in a human-readable format. The seventh step is to drop the 'durations_table' if necessary.

```
1  -- Step 1: Create time to store start and end time
2  CREATE TABLE IF NOT EXISTS durations_table (source TEXT, time_start_stop TEXT);
3
4  -- Step 2: Delete previous data
5  DELETE FROM durations_table;
6
7  -- Step 3: This is the start time
8  INSERT INTO durations_table (source, time_start_stop) VALUES ('start', julianday('now'));
9
10 -- Step 4: Your query
11 -- ##### Design 1 #####
12 SELECT t1.*, t2.*
13 FROM table_1 t1
14 LEFT JOIN table_2 t2
15 ON t1.value_string = t2.value_string
16 WHERE t2.id IS NOT NULL AND t2.value_float >= 1500;
17
18 -- Step 5: This is the end time
19 INSERT INTO durations_table (source, time_start_stop) VALUES ('end', julianday('now'));
20
21 -- Step 6: Display start and end time in human understandable format
22 SELECT source, strftime('%Y-%m-%d %H:%M:%f', time_start_stop) AS date_time FROM durations_table;
23
24 -- Step 7: ONLY IF NECESSARY
25 -- DROP TABLE durations_table;
```

Get data extraction start time

SQL QUERY WITH DATA EXTRACTION DURATION

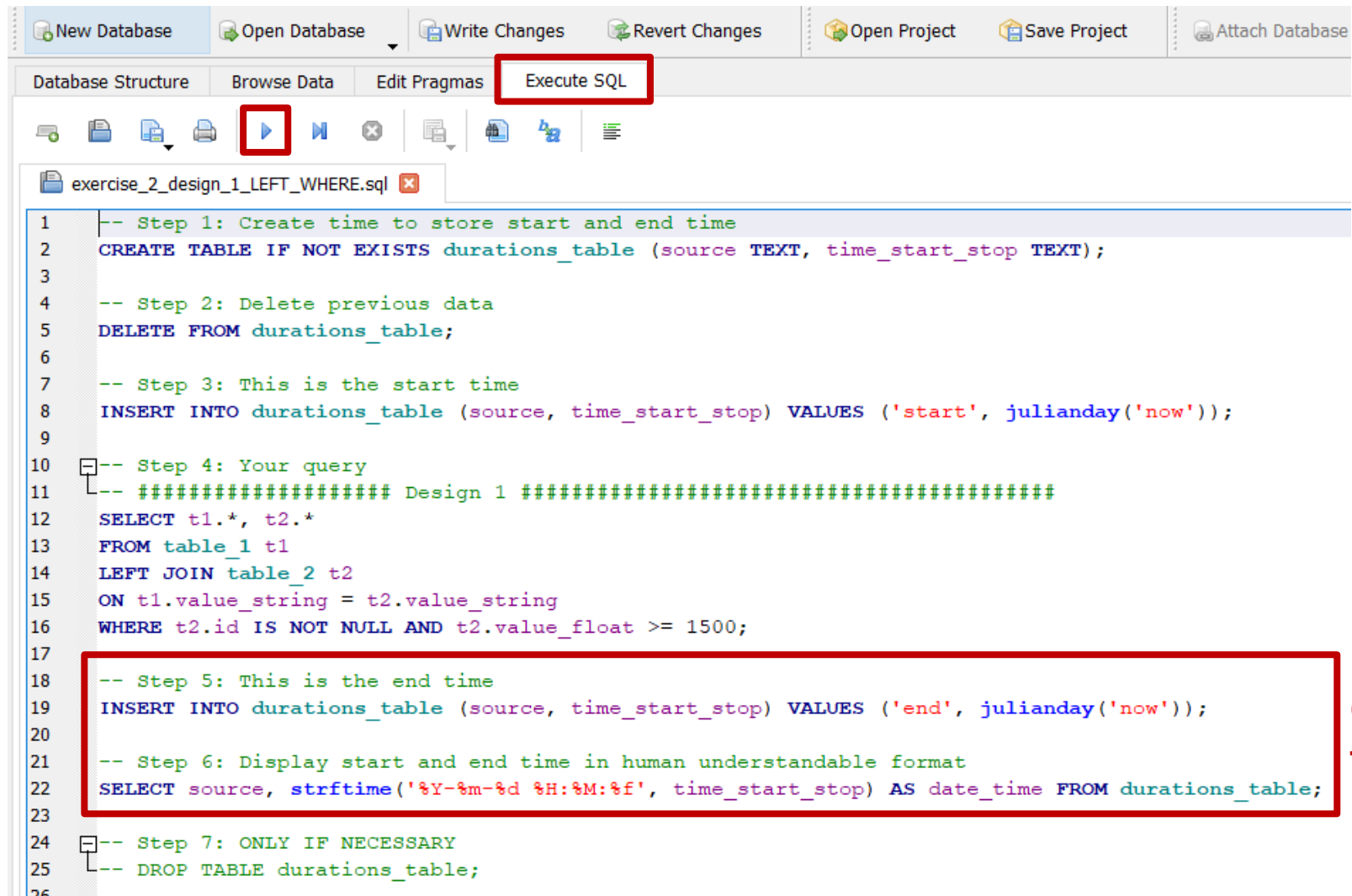


The screenshot shows a database management tool interface. At the top, there are buttons for 'New Database', 'Open Database', 'Write Changes', 'Revert Changes', 'Open Project', 'Save Project', and 'Attach Database'. Below these are tabs for 'Database Structure', 'Browse Data', 'Edit Pragmas', and 'Execute SQL'. The 'Execute SQL' tab is selected and highlighted with a red box. Below the tabs is a toolbar with various icons, including a play button (execute) which is also highlighted with a red box. The main area displays a SQL script in a text editor. The script is titled 'exercise_2_design_1_LEFT_WHERE.sql'. It contains several steps: Step 1: Create a table 'durations_table' with columns 'source' and 'time_start_stop'. Step 2: Delete previous data. Step 3: Insert a start time. Step 4: Execute a query (highlighted with a red box). Step 5: Insert an end time. Step 6: Display the start and end times in a human-readable format. Step 7: Drop the table if necessary.

```
1  -- Step 1: Create time to store start and end time
2  CREATE TABLE IF NOT EXISTS durations_table (source TEXT, time_start_stop TEXT);
3
4  -- Step 2: Delete previous data
5  DELETE FROM durations_table;
6
7  -- Step 3: This is the start time
8  INSERT INTO durations_table (source, time_start_stop) VALUES ('start', julianday('now'));
9
10 -- Step 4: Your query
11 -- ##### Design 1 #####
12 SELECT t1.*, t2.*
13 FROM table_1 t1
14 LEFT JOIN table_2 t2
15 ON t1.value_string = t2.value_string
16 WHERE t2.id IS NOT NULL AND t2.value_float >= 1500;
17
18 -- Step 5: This is the end time
19 INSERT INTO durations_table (source, time_start_stop) VALUES ('end', julianday('now'));
20
21 -- Step 6: Display start and end time in human understandable format
22 SELECT source, strftime('%Y-%m-%d %H:%M:%f', time_start_stop) AS date_time FROM durations_table;
23
24 -- Step 7: ONLY IF NECESSARY
25 DROP TABLE durations_table;
```

Insert your SQL query here

SQL QUERY WITH DATA EXTRACTION DURATION



The screenshot shows a SQL IDE interface with a toolbar at the top containing buttons for 'New Database', 'Open Database', 'Write Changes', 'Revert Changes', 'Open Project', 'Save Project', and 'Attach Database'. Below the toolbar is a tabbed interface with 'Database Structure', 'Browse Data', 'Edit Pragmas', and 'Execute SQL'. The 'Execute SQL' tab is active and highlighted with a red box. Below the tabs is a toolbar with icons for file operations and execution. The 'Execute' icon (a blue play button) is highlighted with a red box. Below the toolbar is a text editor window titled 'exercise_2_design_1_LEFT_WHERE.sql'. The editor contains a SQL script with seven steps, each preceded by a comment. Steps 1, 2, 3, 4, and 5 are grouped by a bracket on the left. Steps 6 and 7 are also grouped by a bracket. The script is as follows:

```
1  -- Step 1: Create time to store start and end time
2  CREATE TABLE IF NOT EXISTS durations_table (source TEXT, time_start_stop TEXT);
3
4  -- Step 2: Delete previous data
5  DELETE FROM durations_table;
6
7  -- Step 3: This is the start time
8  INSERT INTO durations_table (source, time_start_stop) VALUES ('start', julianday('now'));
9
10 -- Step 4: Your query
11 -- ##### Design 1 #####
12 SELECT t1.*, t2.*
13 FROM table_1 t1
14 LEFT JOIN table_2 t2
15 ON t1.value_string = t2.value_string
16 WHERE t2.id IS NOT NULL AND t2.value_float >= 1500;
17
18 -- Step 5: This is the end time
19 INSERT INTO durations_table (source, time_start_stop) VALUES ('end', julianday('now'));
20
21 -- Step 6: Display start and end time in human understandable format
22 SELECT source, strftime('%Y-%m-%d %H:%M:%f', time_start_stop) AS date_time FROM durations_table;
23
24 -- Step 7: ONLY IF NECESSARY
25 -- DROP TABLE durations_table;
```

Get data extraction end time and display both times

Hope you find this useful!



CONNECT WITH ME

CONNECT WITH ME

- LinkedIn

- <https://www.linkedin.com/in/wongchikeongjohn/>

- GitHub

- https://github.com/johnwck/my_da_ds_work/tree/master