

Travel planning for flight data

HIERARCHICAL AND RECURSIVE QUERIES IN SQL SERVER

SQL

Dominik Egarter
Data Engineering Enthusiast

Scoreboard of an airport

| | planning | status | terminal | gate | airline | flight | status |
|------|---------------------|--------|----------|------|---------|--------|--------|
| 804 | SARAJEVO | 16:05 | | | | | |
| 858 | STOCKHOLM-ARLANDA | 16:05 | | | | | |
| 980 | OSLO-GARDERMOEN | 16:05 | | | | | |
| 1132 | DUBLIN | 16:05 | | | | | |
| 1144 | BARCELONA | 16:05 | | | | | |
| 1222 | BILBAO | 16:05 | | | | | |
| 3620 | GENEVE | 16:05 | | | | | |
| 082 | KOELN HBF | 16:05 | | | | | |
| 220 | DUESSELDORF | 16:05 | | | | | |
| 328 | HERINGSBURG | 16:05 | | | | | |
| 112 | VENEDIG-MARCO POLO | 16:10 | | | | | |
| 1368 | MUENCHEN | 16:10 | | | | | |
| 1412 | KRAKAU | 16:10 | | | | | |
| 168 | SPLIT | 16:15 | | | | | |
| 238 | JEDDAH | 16:15 | | | | | |
| 1088 | ROM-FIUMICINO | 16:15 | | | | | |
| 1340 | MARSEILLE | 16:20 | | | | | |
| 964 | BUDAPEST | 16:20 | | | | | |
| 996 | EDINBURGH | 16:20 | | | | | |
| 1040 | AMSTERDAM | 16:25 | | | | | |
| 1196 | PARIS CH. DE GAULLE | 16:25 | | | | | |
| 284 | ZUERICH | 16:25 | | | | | |

How is a flight data set structured?

| Departure | Arrival | FlightNumber | Cost | Time |
|-----------|----------|--------------|------|------|
| London | Paris | LH3827 | 90 | 2 |
| Vienna | New York | MH2370 | 379 | 8 |
| New York | Paris | LH9832 | 489 | 9 |
| Vienna | Paris | SU2389 | 200 | 3 |
| London | Chicago | OP1230 | 650 | 10 |
| New York | Chicago | NL5460 | 150 | 2 |

How to build a flight route?



- Use recursion to get **all possible** flight routes
- A route is defined by the **departure** airport and the **destination** airport
- Limit the number of possible layovers to create realistic flight routes

Building a flight route - step 1

```
WITH flightRoute (Departure, Arrival, stops) AS(  
  -- Anchor query  
  SELECT f.Departure, f.Arrival, 0  
    FROM flightPlan f  
   WHERE Departure = 'Vienna'  
  -- Recursive query  
  UNION ALL  
    SELECT p.Departure, f.Arrival, p.stops + 1  
    FROM flightPlan f, flightRoute p  
   WHERE p.Arrival = f.Departure AND  
         p.stops < 5  
)
```

```
SELECT Departure, Arrival, stops  
FROM flightRoute
```

| Departure | Arrival | stops |
|-----------|---------------|-------|
| Vienna | Paris | 2 |
| Vienna | San Francisco | 3 |
| Vienna | Vienna | 3 |
| Vienna | Frankfurt | 3 |
| ... | ... | ... |

Building a flight route - step 2

```
WITH flightRoute (Departure, Arrival, stops, route) AS(  
  SELECT f.Departure, f.Arrival, 0,  
  CAST(Departure + '->' + Arrival AS VARCHAR(MAX))  
  FROM flightPlan f  
  WHERE Departure = 'Vienna'  
  UNION ALL  
  SELECT p.Departure, f.Arrival, p.stops + 1,  
  p.totalCost + f.Cost,  
  CAST(p.route + '->' + f.Arrival AS VARCHAR(MAX))  
  FROM flightPlan f, flightRoute p  
  WHERE p.Arrival = f.Departure AND p.stops < 5  
)
```

- Introduce `route` in the anchor member
- Track `route` s in recursive member
- Limit the number of stops

Building a flight route - result

```
SELECT Departure, Arrival, Route
FROM flightRoute
```

```
+-----+-----+-----+
| Departure | Arrival | route |
|-----|-----|-----+
| London    | New York | London -> Vienna -> Chicago -> New York |
| Vienna    | Chicago  | Vienna -> London -> Chicago |
| Paris     | Los Angeles | Paris -> Toronto -> Los Angeles |
| Chicago   | New York | Chicago -> New York |
| Rome      | New York | Rome -> London -> Chicago -> New York |
| ...       | ...      | ... |
+-----+-----+-----+
```

Querying for possible flight with limits

```
WITH flightRoute (Departure, Arrival, stops, totalCost, route) AS(
  SELECT f.Departure, f.Arrival, 0, Cost,
    CAST(Departure + '->' + Arrival AS NVARCHAR(MAX))
    FROM flightPlan f
    WHERE Departure = 'New York'
  UNION ALL
  SELECT p.Departure, f.Arrival, p.stops+1,
    p.totalCost + f.Cost, p.route + '->' + f.Arrival
    FROM flightPlan f, flightRoute p
    WHERE p.Arrival = f.Departure AND p.stops < '...'
)
```

```
SELECT '...'
FROM flightRoute
WHERE '...' ;
```

Find all possible destination airports where:

- The departure airport is fixed
 - New York
- The number of stops is limited to n
- The output is limited by a condition
 - cost limit
 - cheapest route to some destination

Let's find possible flight routes!

HIERARCHICAL AND RECURSIVE QUERIES IN SQL SERVER

How to assemble a car?

HIERARCHICAL AND RECURSIVE QUERIES IN SQL SERVER



Dominik Egarter
Data Engineering Enthusiast

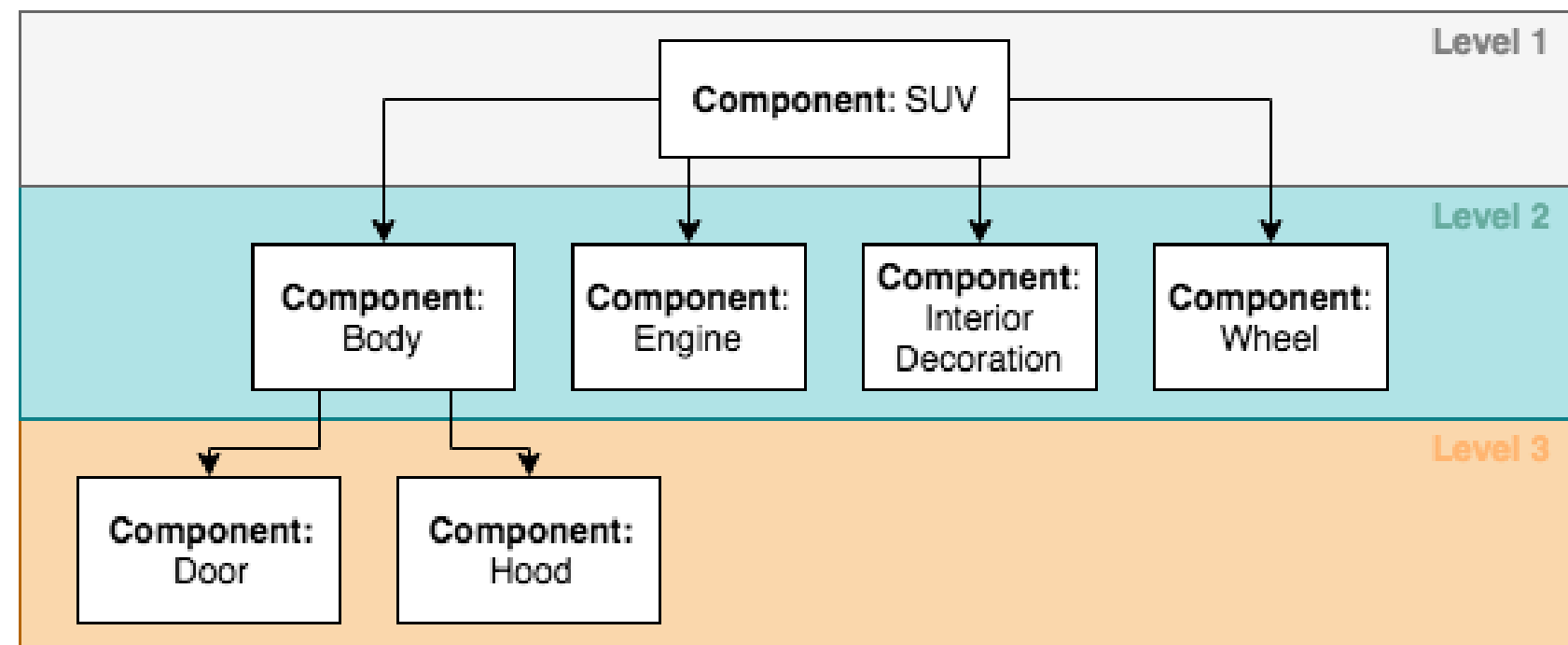
Disassemble a car



List of parts of a car

Different levels of components:

- **Level 1:** *SUV, Cabrio*
- **Level 2:** *Body, Engine, Interior Decoration, Wheel*
- **Level 3:** *Door, Hood, Engine Body, Cylinder, Seats*



Create the data model

Elements to create **hierarchy**:

- `PartID` & `SubPartID`

Elements to describe **characteristics**:

- `Component` : *Engine*
- `Title` : *V6BiTurbo*
- `Vendor` : *BMW*
- `ProductKey` : *EV3891ASF*
- `Cost` : *3000*
- `Quantity` : *1*

| BillOfMaterial |
|---------------------------|
| + PartID: INT primary key |
| + SubPartID: INT |
| + Component: VARCHAR(255) |
| + Title: VARCHAR(255) |
| + Vendor: VARCHAR(255) |
| + ProductKey: CHAR(32) |
| + Cost: INT |
| + Quantity: INT |

Use the hierarchical data model

- What are the levels of components that build up a car?

```
+-----+-----+
| Component | Level |
+-----+-----+
| SUV      | 1     |
+-----+-----+
| Body     | 2     |
+-----+-----+
| Hood     | 3     |
+-----+-----+
```

Use the hierarchical data model

- What is the total quantity of each component required to build the car for each component level?

| Component | Quantity |
|-----------|----------|
| SUV | 1 |
| Body | 1 |
| Wheels | 4 |

Let's assemble a car!

HIERARCHICAL AND RECURSIVE QUERIES IN SQL SERVER

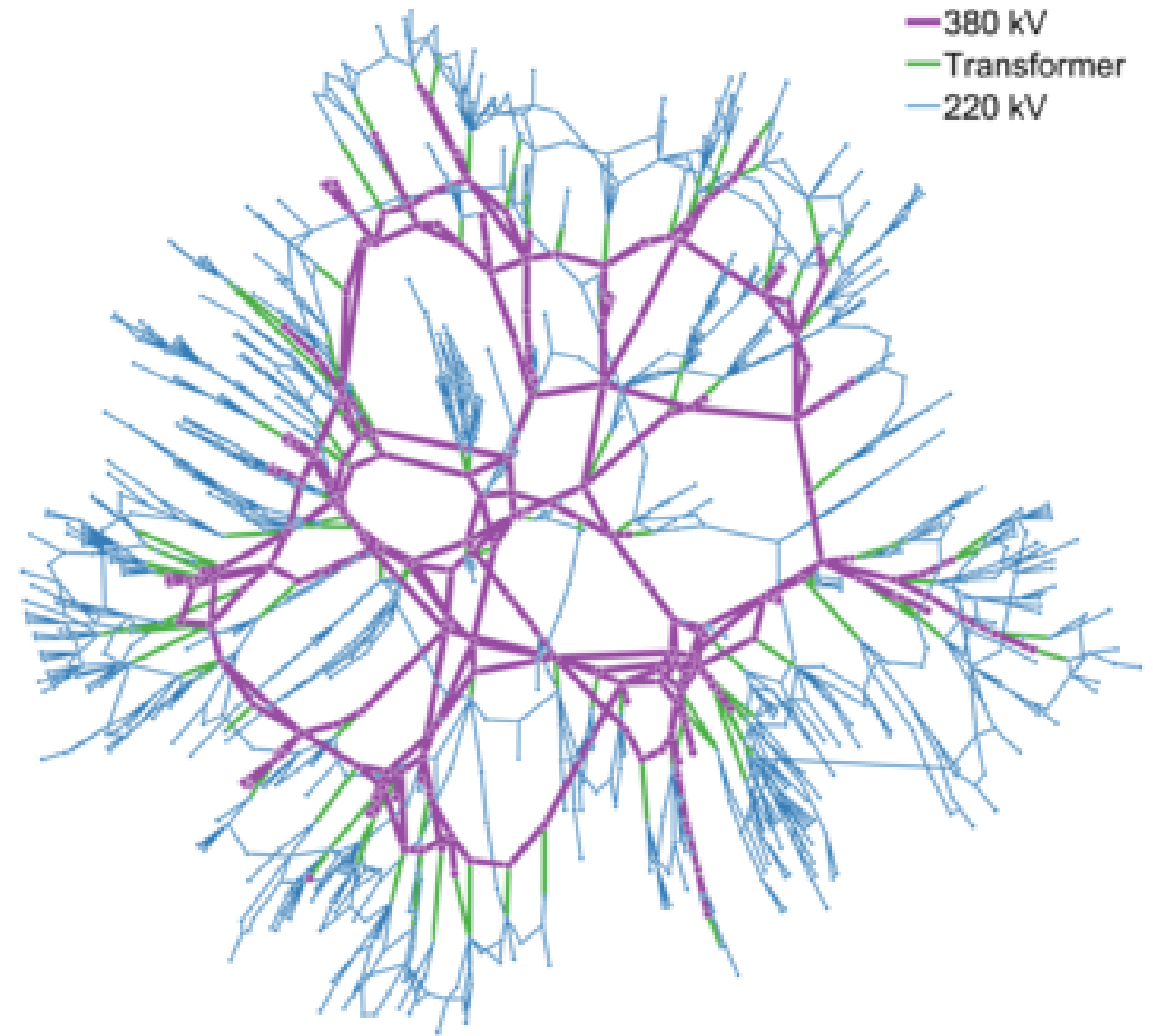
Modeling a power grid

HIERARCHICAL AND RECURSIVE QUERIES IN SQL SERVER



Dominik Egarter
Data Engineering Enthusiast

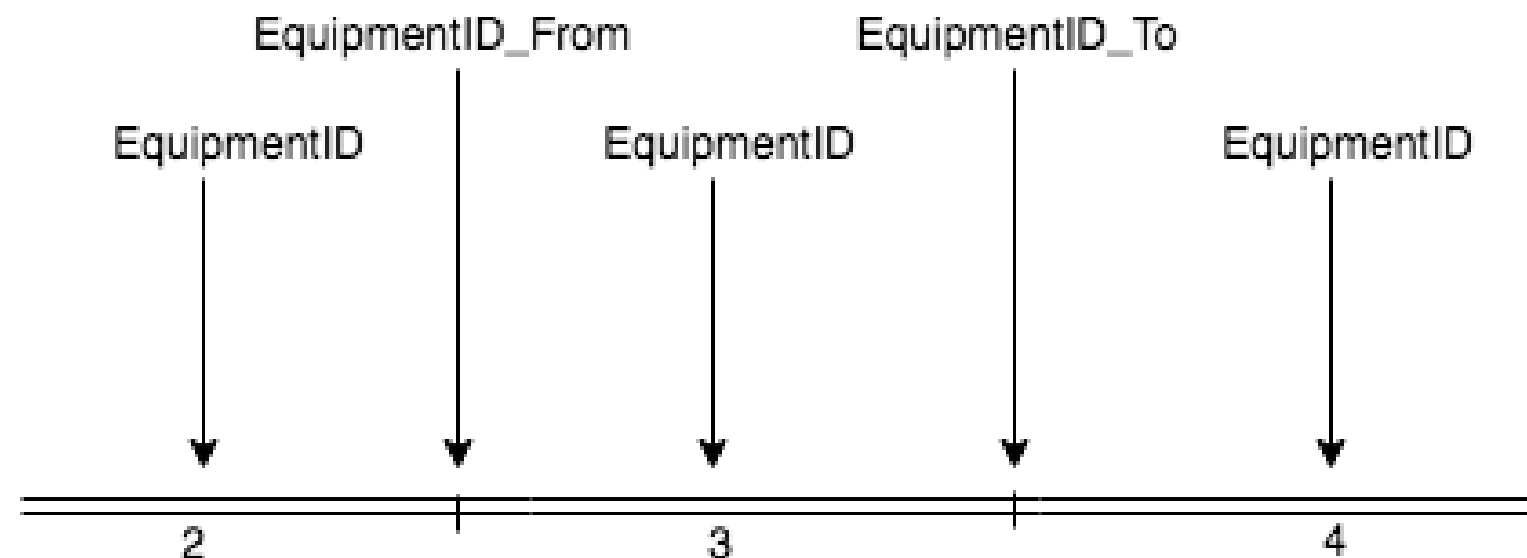
The power grid



Modeling a power grid

You need three ID values:

- ID of the power line: `EquipmentID`
- ID of the first connected power line: `EquipmentID_From`
- ID of the second connected power line: `EquipmentID_To`



Characteristics of power lines

- **Voltage Level**

HV - high Voltage, MV - medium voltage, LV - low voltage

- **Description**

Cable, Overhead Line, Transformer

- **Construction Year:** Year of construction

- **Inspection Year:** Year of the last inspection

- **Condition Assessment:**

good, bad, repair, exchange

Common task for grid maintenance

Find the power lines to be replaced

- *Find the power lines that are connected to each other: use recursion to find the connected power lines*
- *Find power lines with bad, exchange or repair condition*

```
+-----+-----+
| Line   | Condition |
+-----+-----+
| 1      | exchange  |
+-----+-----+
| 2      | repair    |
+-----+-----+
| 3      | bad       |
+-----+-----+
```



Let's find the power lines to be maintained!

HIERARCHICAL AND RECURSIVE QUERIES IN SQL SERVER

Summary of the course

HIERARCHICAL AND RECURSIVE QUERIES IN SQL SERVER



Dominik Egarter
Data Engineering Enthusiast

Chapter 1: Recursion and CTEs

What is recursion?



Recursion is the use of a procedure, subroutine, function, or algorithm that calls itself one or more times until a specified condition is met

Definition of a Common Table Expression (CTE):

```
WITH CTEtable as (  
    <select statement on a table>  
)  
  
SELECT *  
FROM CTEtable
```

Specifies a temporary named result set, known as a common table expression (CTE)

Chapter 2: Hierarchical and recursive queries

Definition of a recursive CTE:

```
WITH cte_name AS (  
    -- Anchor member  
    <cte_initial_query>  
    UNION ALL  
    -- Recursive member  
    <cte_recursive_query> )  
  
SELECT *  
FROM cte_name
```

Real-world examples:

1. Mathematical problems
2. Hierarchy of an organization
3. Hierarchy of a family tree

Chapter 3: Creating data models on your own

Manipulating a table:

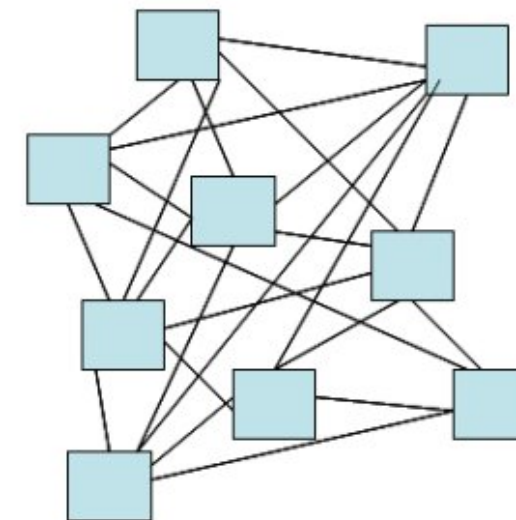
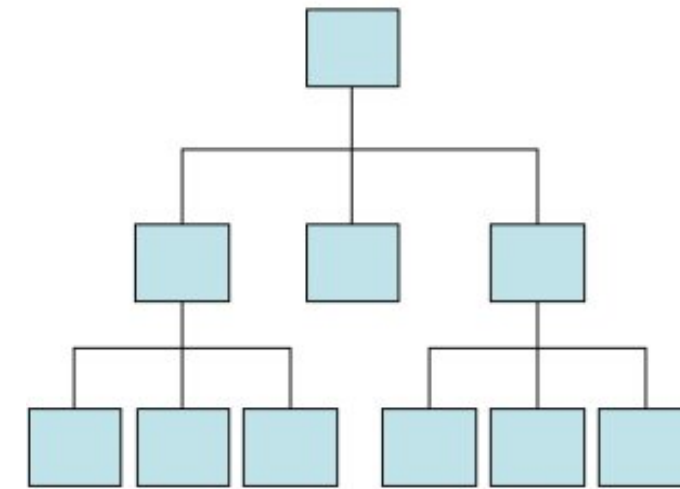
- `CREATE` , `INSERT` , `ALTER` , `DROP`

Relational data model:

- The relational database model is the most widely used database model.

Hierarchical and networked data model:

- Represented as tree structure
- Has one (hierarchy) or many (networked) root element



Chapter 4: Hierarchical queries of real world examples

Common tasks:

- Create a hierarchy data model
- Query the hierarchy recursively
- Get the level of a hierarchy

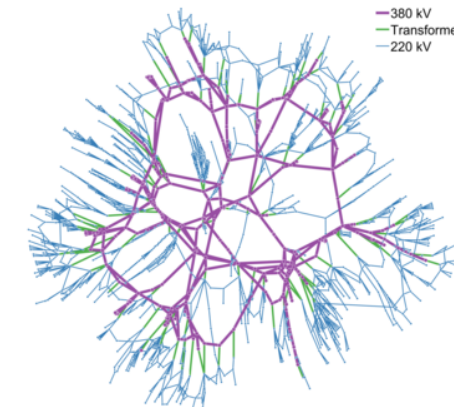
How to assemble a car?



Travel planning of flight data:

A large flight information display board in an airport terminal, showing flight schedules for various destinations. The board is divided into sections for different airlines and destinations, with columns for flight number, destination, departure time, and gate. The destinations listed include Stockholm-Arlanda, Oslo-Gardermoen, Barcelona, Bilbao, Genoa, Cologne HBF, Dusseldorf, Heringsdorf, Venedig-Marco Polo, Muenchen, Krakau, Split, Jeddah, Rom-Fiumicino, Marseille, Budapest, Edinburgh, Amsterdam, Paris Ch. de Gaulle, Zurich, and Innsbruck. The board is illuminated with bright lights, and the background shows the airport's architecture.

Modeling a power grid



Congratulations!

HIERARCHICAL AND RECURSIVE QUERIES IN SQL SERVER