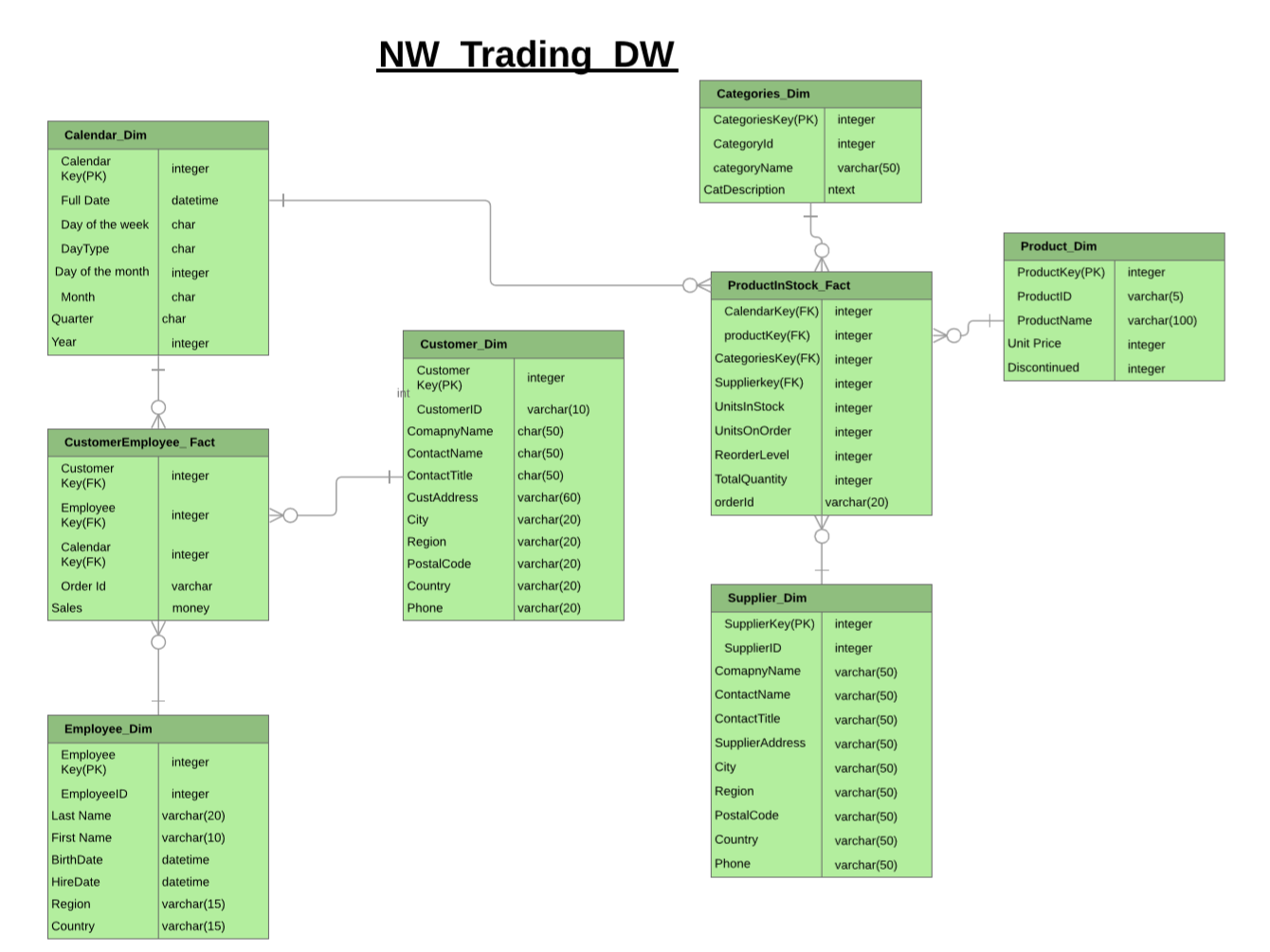
**NEO4J**

**The schema for my warehouse is:**



I have implemented the CustomerEmployee\_Fact with Calendar\_dim, Employee\_Dim and Customer\_Dim Dimensions using Neo4j.

The following are the CQL codes used to generate the graphs in Neo4j:

**1. Loading data from CSV files:**

**In this step the data is extracted from the CSV files stored in Neo4j’s import folder. We also create constraints on each dimension.** **Constraints are created to define Primary keys in each Dimension.**

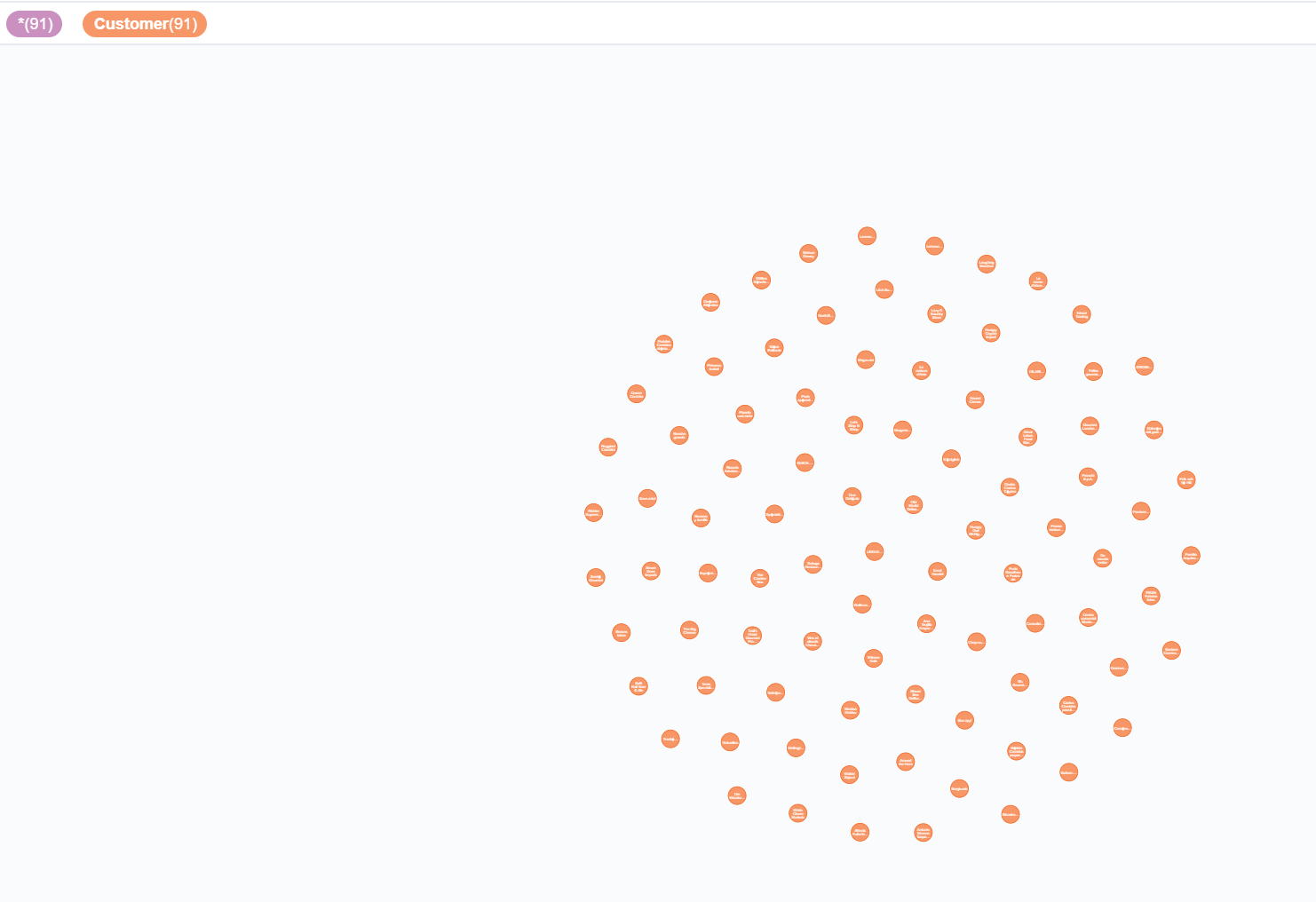
* 1. **Loading the Customer CSV file:**

load csv with headers from "file:///CustomerDim.csv" as row create(c:Customer) set c={CustomerKey:row.CustomerKey, CustomerID:row.CustomerID, CompanyName:row.CompanyName,ContactName:row.ContactName,ContactTitle:row.ContactTitle, CustAddress:row.CustAddress,City:row.City, Region:row.Region, PostalCode:row.PostalCode,Country:row.Country,Phone:row.Phone }

return c

**Creating Constraint for it:**

CREATE CONSTRAINT ON (c:Customer) ASSERT c.CustomerID is unique



**1.2 Loading the CSV file Employee:**

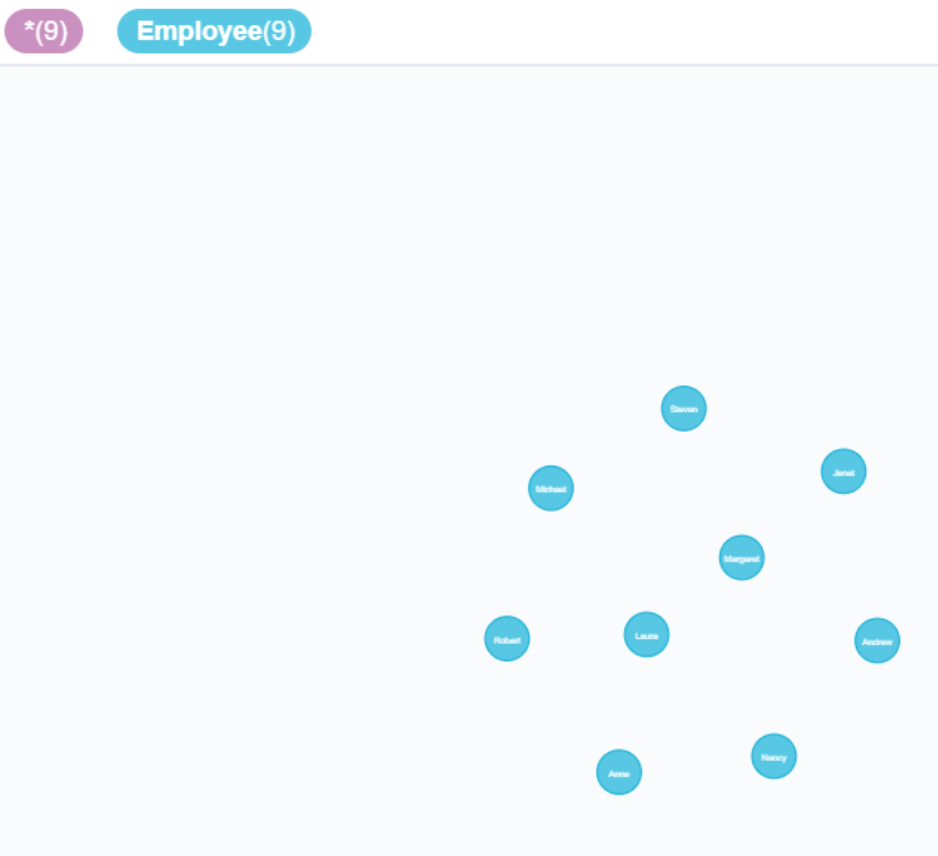
load csv with headers from "file:///EmployeeDim.csv" as row create(e:Employee) set e={ EmployeeKey:row. EmployeeKey, EmployeeID:row.EmployeeID, LastName:row.LastName, FirstName:row.FirstName,

BirthDate:row.BirthDate, HireDate:row. HireDate, Region:row.Region, Country:row.Country }

return e

**Creating Constraint for it:**

CREATE CONSTRAINT ON (e:Employee) ASSERT e.EmployeeID is unique



**1.3 Loading the CSV file Calendar:**

load csv with headers from "file:///CalendarDim.csv" as row create(cl:Calendar) set cl={ CalendarKey:row.CalendarKey, FullDate:row.FullDate,

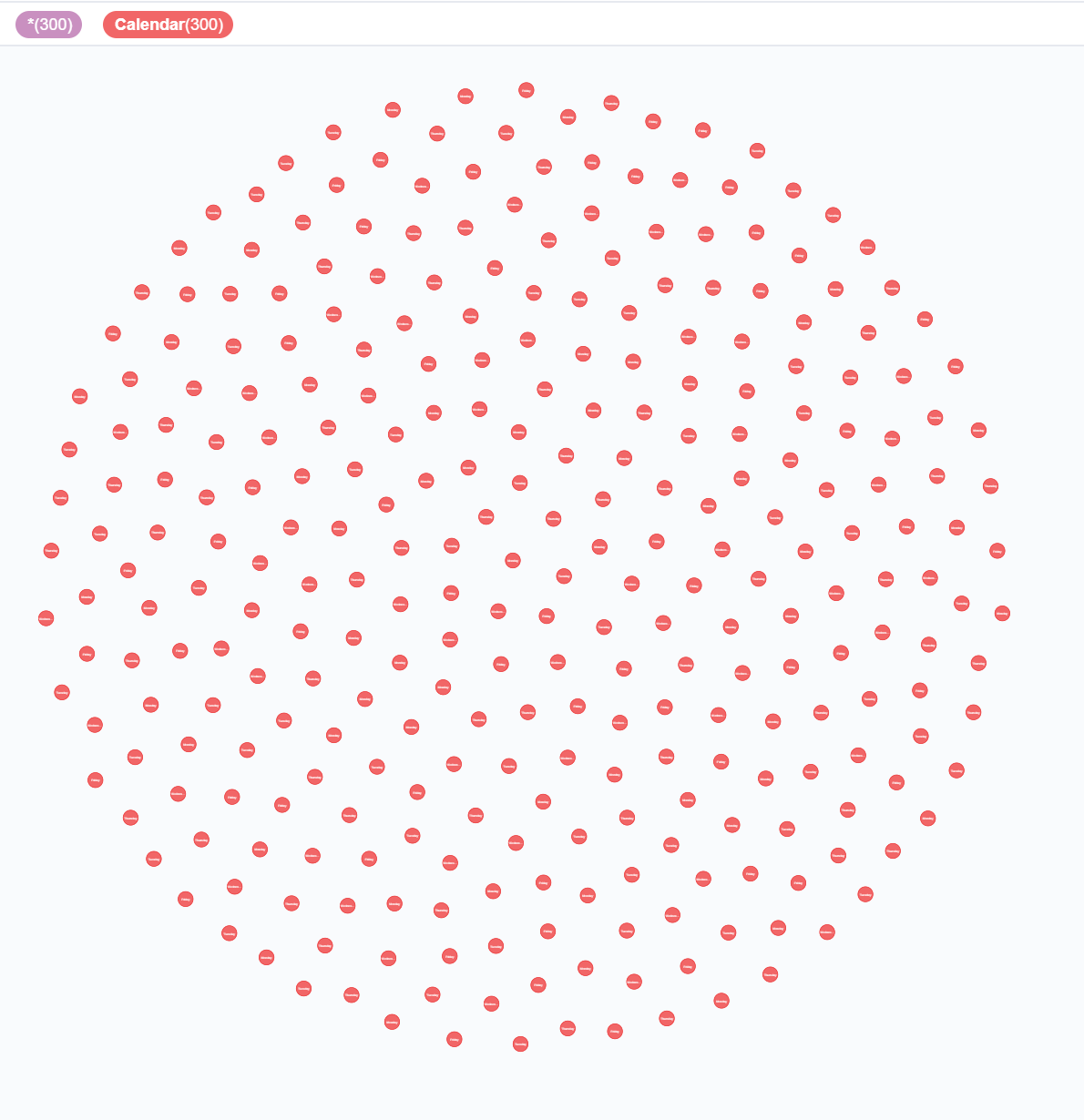
DayofWeek\_:row.DayofWeek\_, DayType:row.DayType,

DayofMonth\_:row.DayofMonth\_, Month\_:row.Month\_, Quarter\_:row. Quarter\_, Year\_:row. Year\_}

return cl

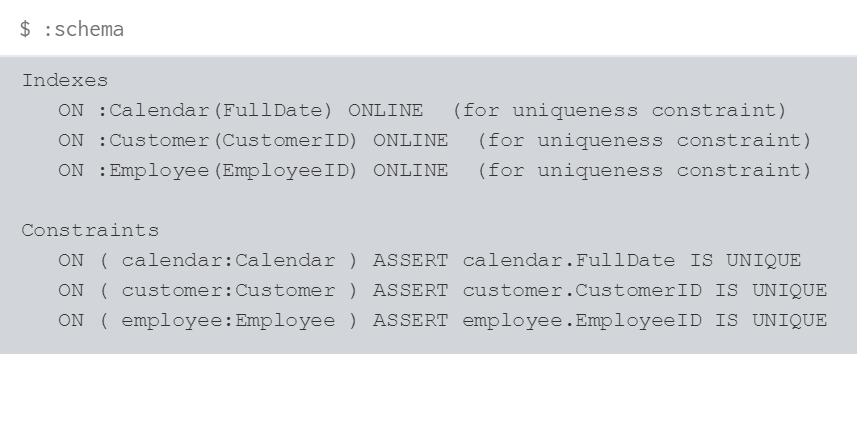
**Creating Constraint for it:**

CREATE CONSTRAINT ON (cl:Calendar) ASSERT cl.FullDate is unique



**To recheck the constraints**

:schema



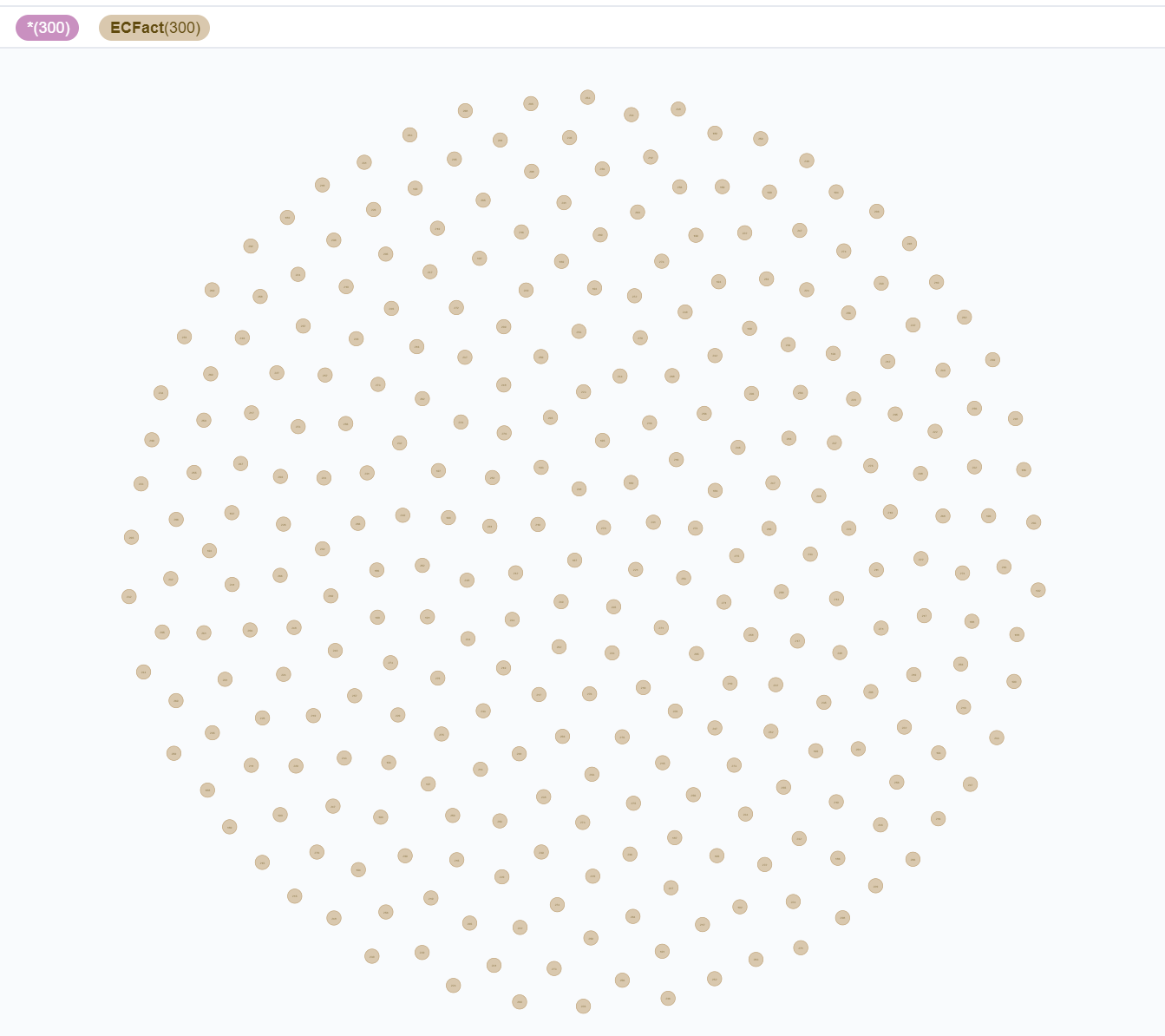
**1.4 Loading the CSV file EmployeeCustomerFact**

load csv with headers from "file:///EmployeeCustFact.csv" as row create(ecf:ECFact) set ecf={ customerkey:row.customerkey, Employeekey:row.Employeekey,

calendarkey:row.calendarkey, orderid:row.orderid,

Sales:row.Sales}

return ecf

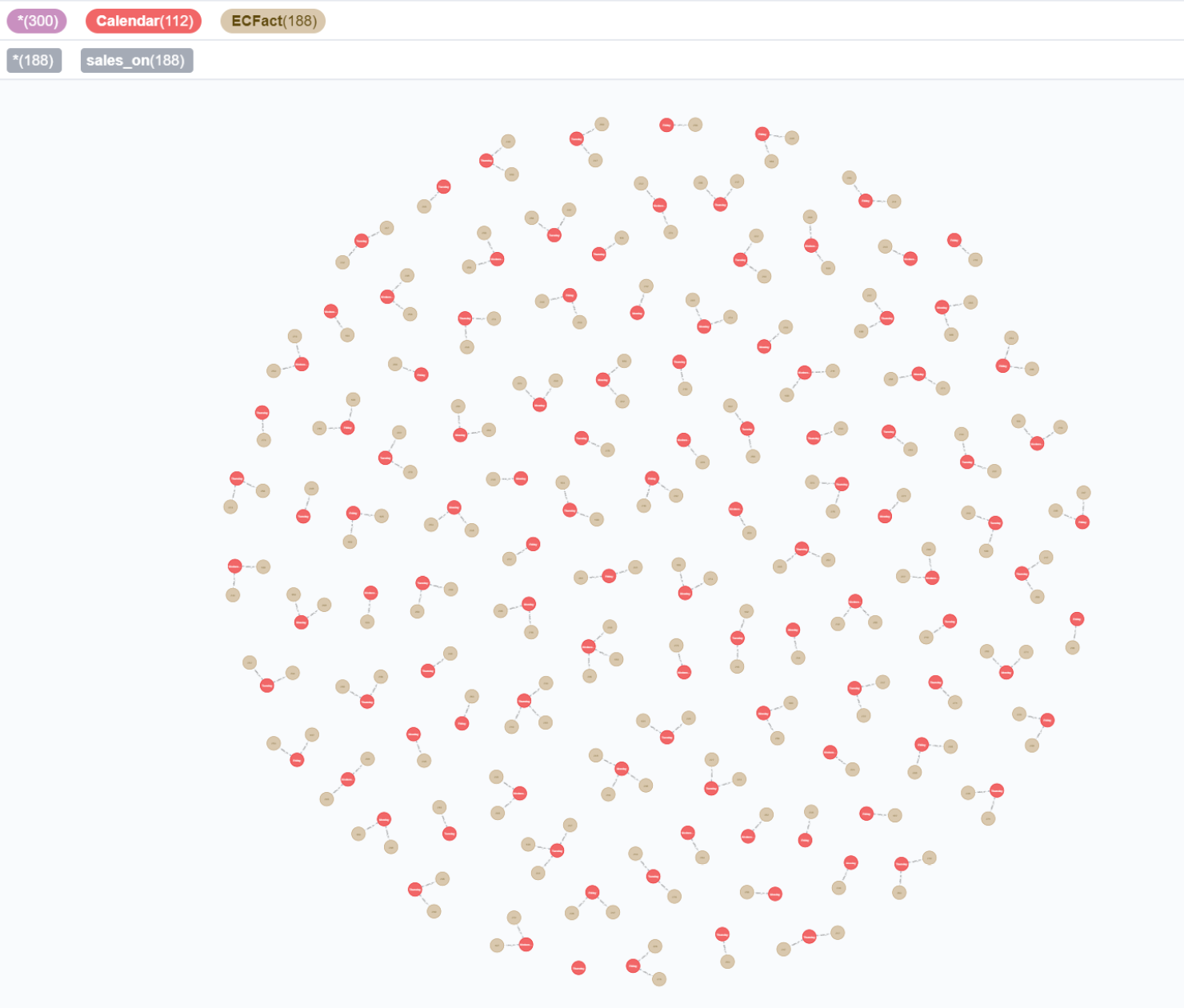


**2. Creating Relationships between entities:**

**To connect the fact(CustomerEmployee\_Fact) to each Dimension.**

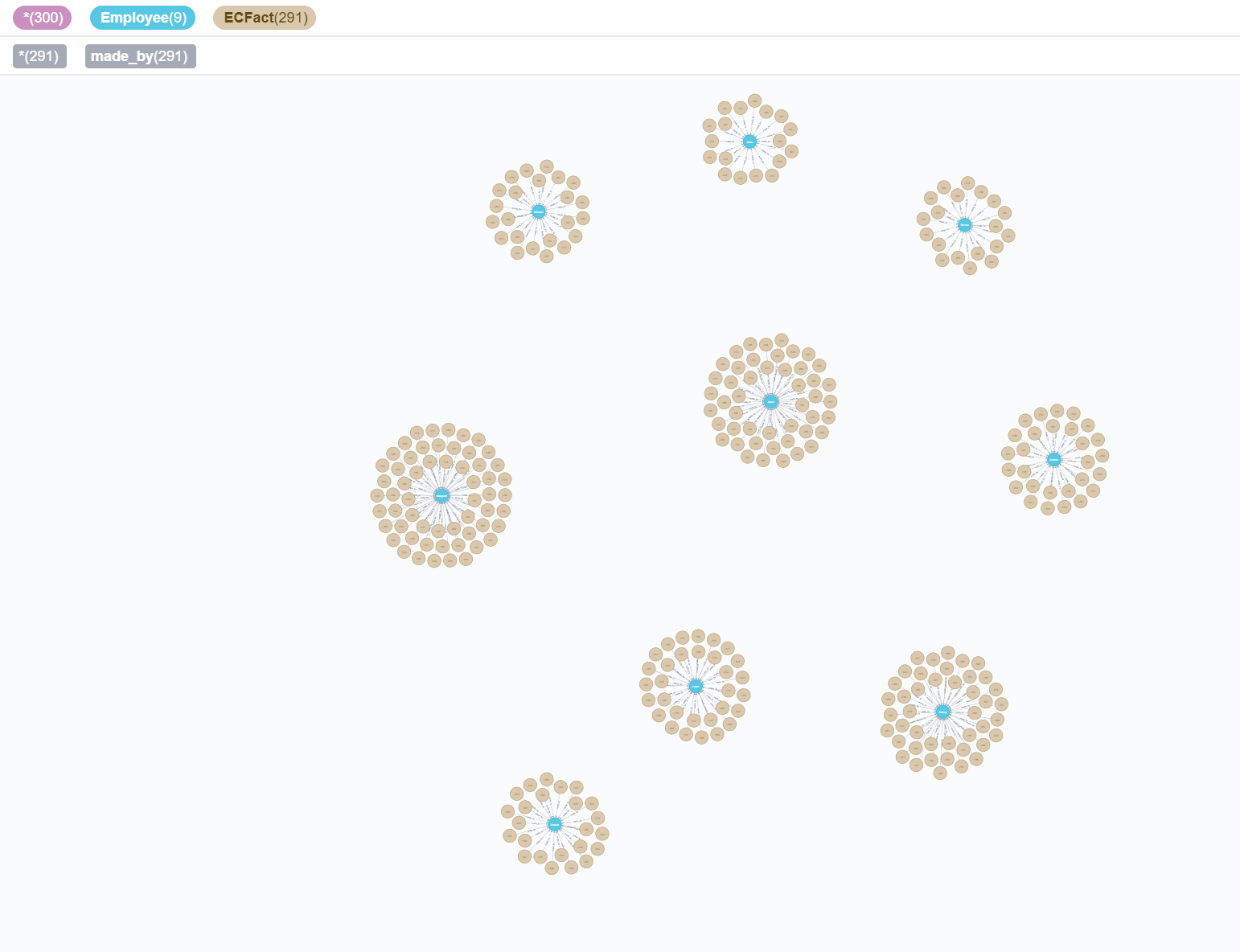
**Calendar:**

match (ecf:ECFact), (cl:Calendar) where ecf.calendarkey = cl.CalendarKey create (ecf)-[r:sales\_on] ->(cl) return cl,ecf,r



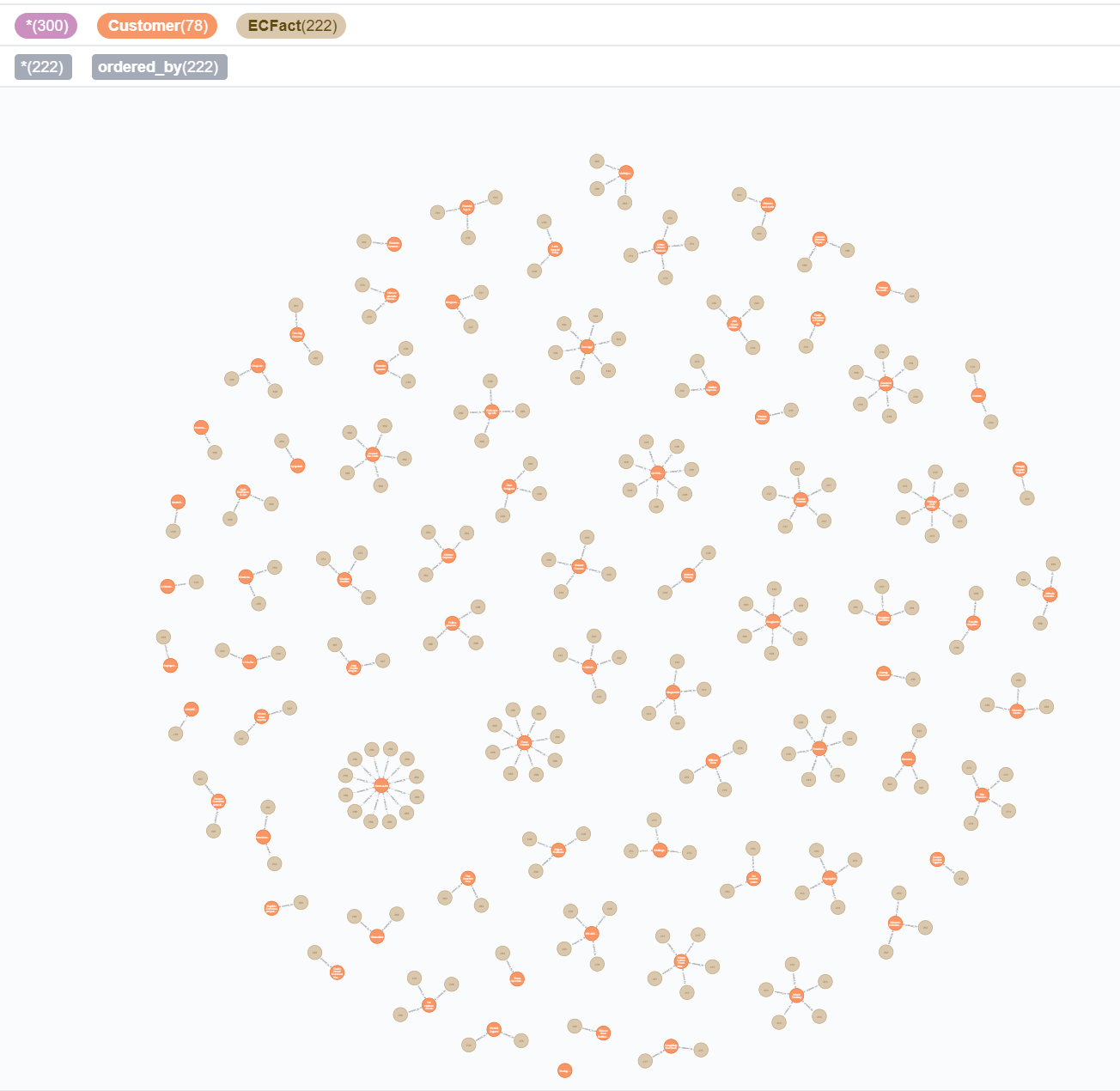
**Employee:**

match (ecf:ECFact), (e:Employee) where ecf.Employeekey = e.EmployeeKey create (ecf)-[r:made\_by] ->(e) return e,ecf,r



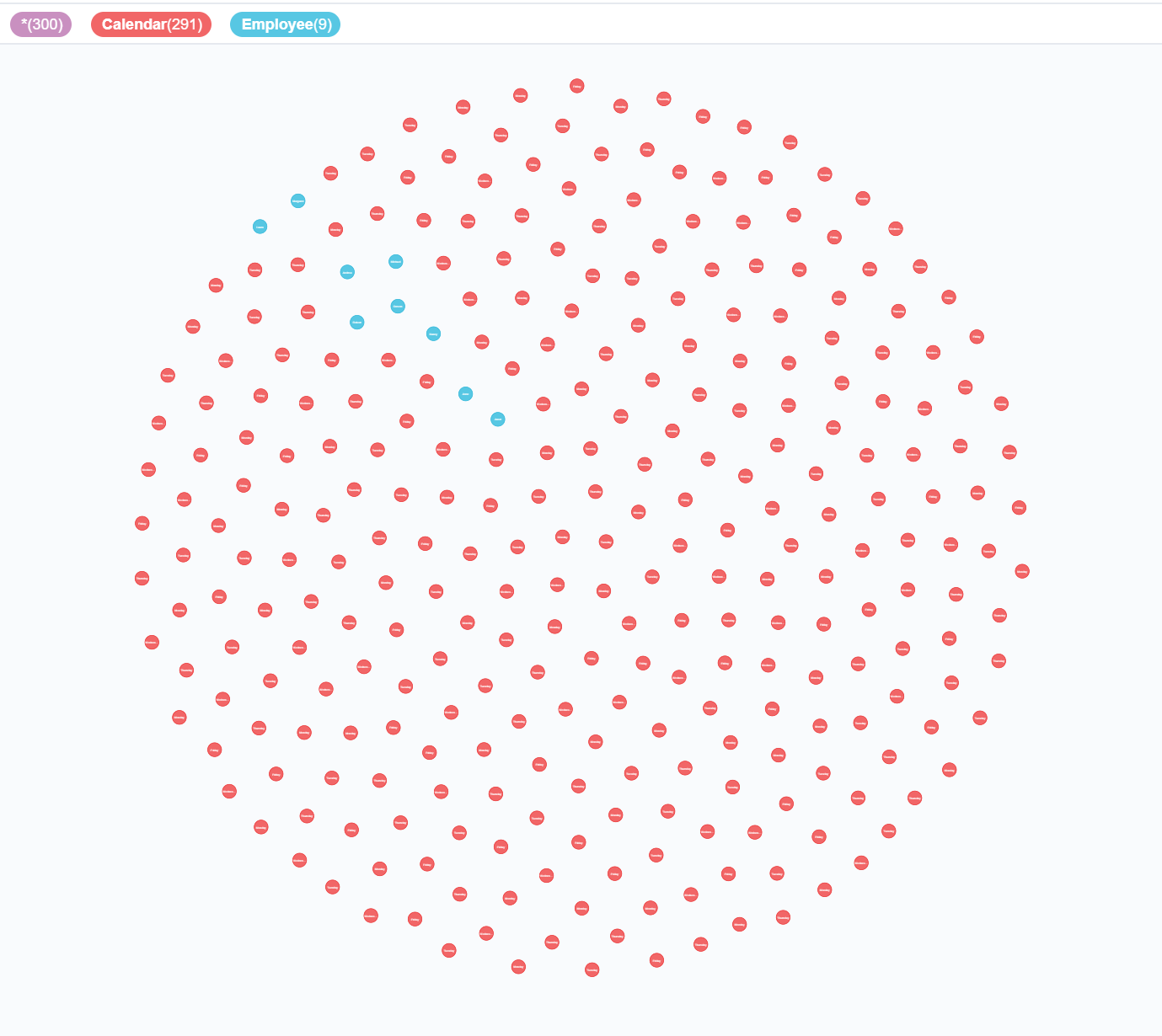
**Customer:**

match (ecf:ECFact), (c:Customer) where ecf.customerkey = c.CustomerKey create (ecf)-[r:ordered\_by] ->(c) return c,ecf,r



**Checking overall structure:**

match(n) return n



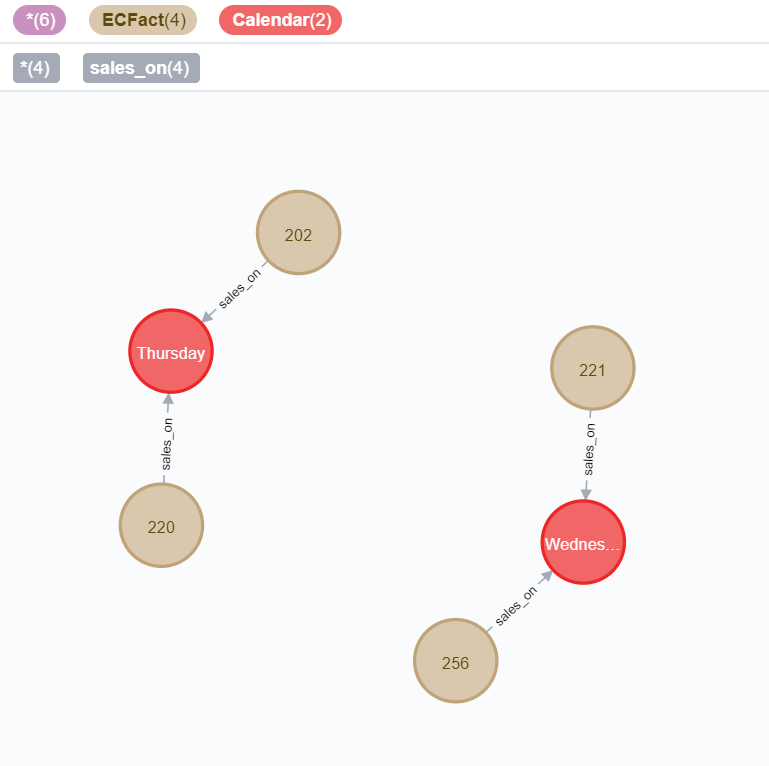
**3. Output Queries:**

**The output can be graphical or tabular.**

**3.1 Sales on specific date of each year:**

**This query fetches the sales on 25th December of each year(1996-1997) from the data warehouse.**

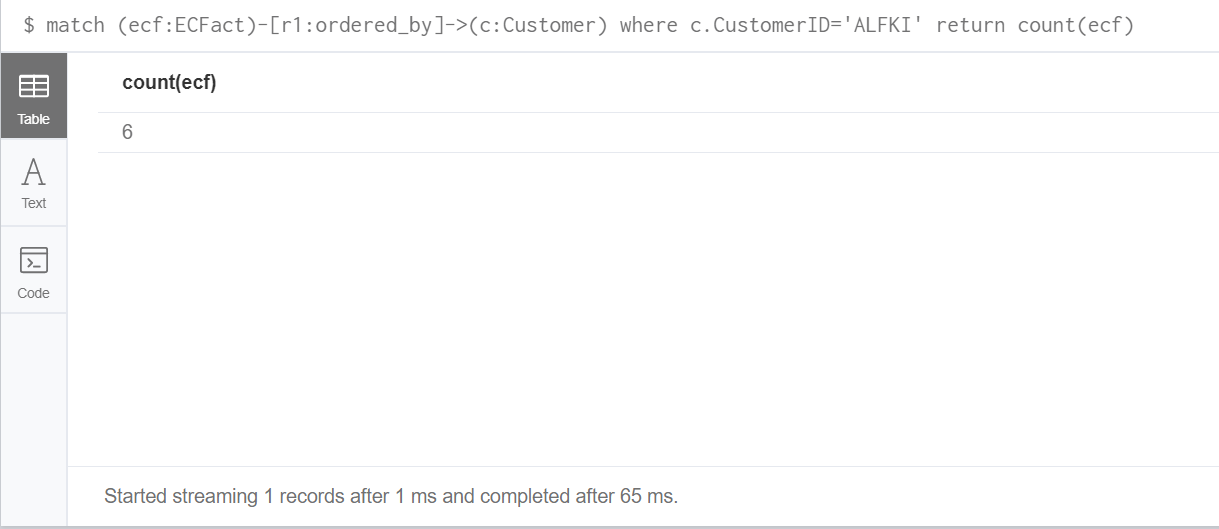
match (ecf:ECFact)-[r:sales\_on]->(cl:Calendar) where cl.Month\_='12' AND cl.DayofMonth\_='25' return ecf,cl,r



**3.2 Count of entries for a given Customer:**

**This query fetches the order count of customer with customer id ALFKI.**

match (ecf:ECFact)-[r1:ordered\_by]->(c:Customer) where c.CustomerID='ALFKI' return count(ecf)

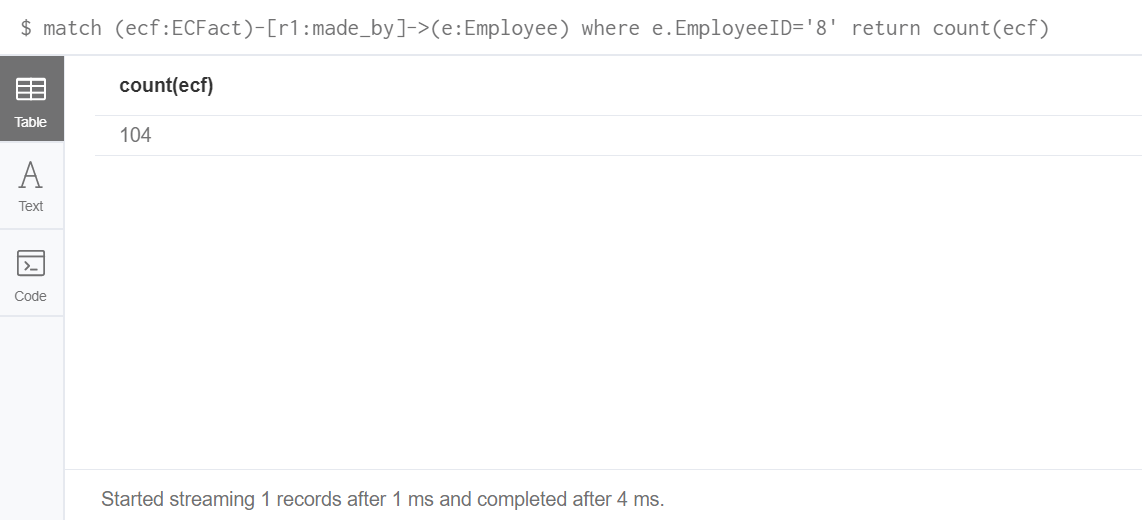


count -- 6

**3.3 Employee sales count:**

**This query fetches no of orders punched by EmployeeID= 8.**

match (ecf:ECFact)-[r1:made\_by]->(e:Employee) where e.EmployeeID='8' return count(ecf)



**3.4 Employee – Sales Relationship:**

**This query maps the sales of each employee.**

match (ecf:ECFact)-[r1:made\_by]->(e:Employee) return ecf,e,r1

