# Fysiek ontwerp

We hebben ervoor gekozen om 2 secundaire indexen aan te maken op de tabel Employee. Deze tabel hebben we gekozen omdat we van mening zijn dat hij niet vaak worden aangepast en door veel query’s gebruikt wordt.

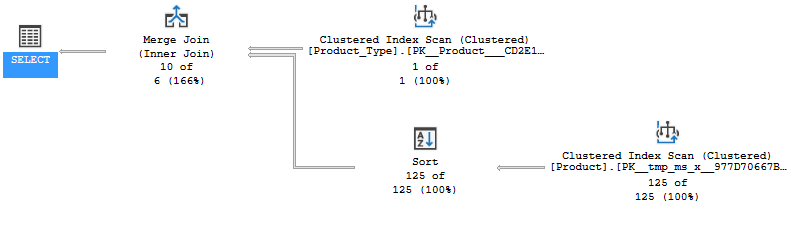
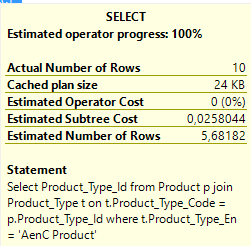
## Product

De tabel product is een tabel waarvan heel vaak data opgevraagd wordt. Ook heeft de tabel nu 125 records. We hebben daarom besloten een secundaire index toe te voegen op de kolommen Product\_Type en Manager\_Id. Deze kolommen hebben we gekozen aangezien product type iets is wat mensen vaak opzoeken. En Manager\_Id aangezien er niet vaak managers bij komen en wij verwachten dat er veel query’s zijn die van deze index profiteren.

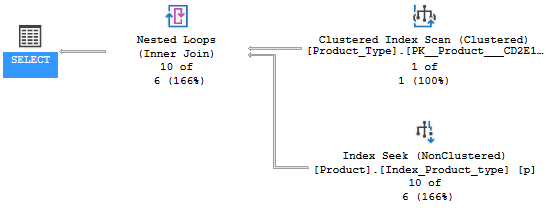
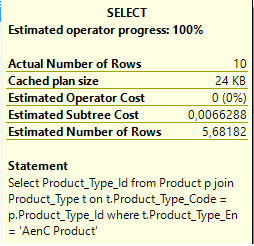
### Preformance test Product\_Type

C:\Users\Cas Ros\Documents\HHS\jaar 2\periode 3\Project\Project-jaar-2-periode-3\fysiek ontwerp afb\productItemIndex1.PNG

We hebben de performance van de index getest met behulp van de bovenstaande query. Als resultaat kwam het onderstaande execution plan.

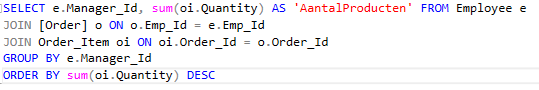


Uit het execution plan is niet veel informatie te herleiden aangezien de costen 100% of 166% zijn.

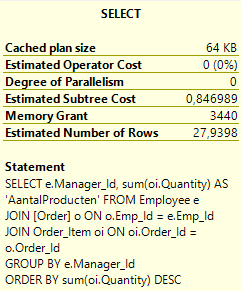
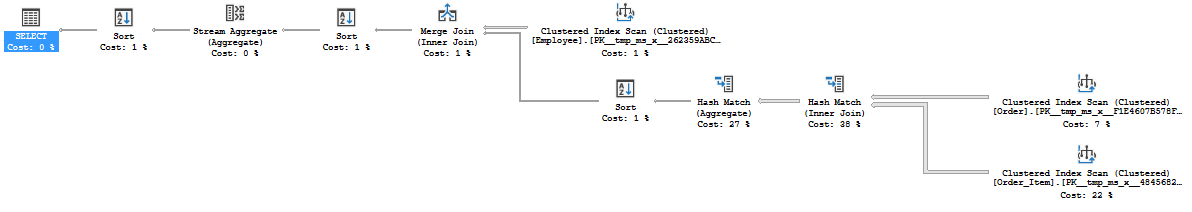


door de executies van deze query voor en na het aanmaken van de index te vergelijken zien we dat de subtree cost aanzienlijk is vermindert. Ook kan je aan het execution plan zien dat hij gebruik heeft gemaakt van de aangemaakte index. Het probleem met de kosten van 100% en 166% is er nog steeds, maar nu is er 1 stap minder in de execution planner.

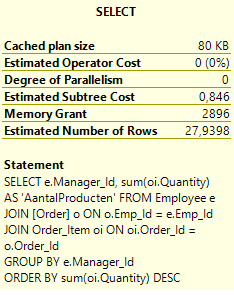
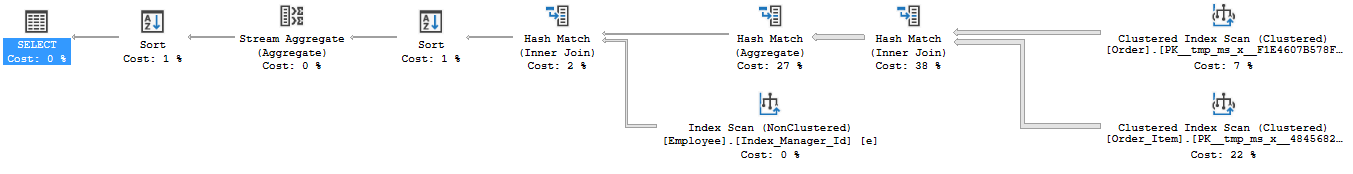
### Preformance test Manager\_Id



We hebben de performance van de index getest met behulp van de bovenstaande query. Als resultaat kwam het onderstaande execution plan.



Hierin is te zien dat de execution planner maar 1% nodig heeft voor de employee tabel. Toch hebben we ervoor gekozen hier de index op te doen aangezien de Order en Order\_Item tabellen zeer vaak met nieuwe data gevuld worden, waardoor indexen op deze tabellen constant bij gewerkt zouden moeten worden.



Ook bij dit execution plan is te zien dat na het aanmaken van de index de query hiervan gebruik maakt. Als resultaat heeft de query nu minder memory nodig. Ook is de cost die eerst 1% was nu 0%.