1 Subqueries

1.1 Inleiding

In SQL kun je meestal op verschillende manieren een zelfde resultaat verkrijgen. Subqueries zijn voor de voorbeelden die volgen niet noodzakelijk de enige weg. De voorbeelden leggen enkel het principe en de werking van subqueries uit.

Query's kunnen erg complex en lang zijn en veel schrijfwerk vereisen. Zo verlies je gemakkelijk het overzicht. Geneste queries: queries die op de resultatenset van een andere query worden uitgevoerd, kunnen soms een oplossing bieden omdat ze toelaten het probleem in logische stukken te laten uiteenvallen. Je kan voor deze voorbeelden vaak ook een JOIN gebruiken maar werken met subqueries is vaak eenvoudiger.

Voorbeeld 1

Om de adresgegevens te vinden van de klanten die een order plaatsten in juni 2005 moeten volgende vragen worden beantwoord:

- Welke klanten plaatsten een order in juni 2005?
- Wat zijn de adresgegevens van deze klanten?

Het antwoord op de **eerste** vraag vind je in de onderstaande SQL-code, die enkel de klantnummers bevat van de orders geplaatst in juni 2005. Je neemt enkel unieke waarden op. Je bent enkel geïnteresseerd in de primaire sleutel die in het klantenbestand zal linken naar de klantgegevens. Je maakt een view met deze code.

orders juni 2005

CREATE VIEW *'orders juni 2025' as*
SELECT DISTINCT *Klantnummer*
FROM *tblOrders*
WHERE *(MONTH(Orderdatum) = 6) AND (YEAR(Orderdatum) = 2005)*

Het antwoord op de **tweede** vraag vind je in onderstaande query die de passende gegevens haalt uit, enerzijds de view 'orders juni 2005' en anderzijds tblKlanten. Hier spreekt men van geneste query's.

SQL-instructie:

SELECT *a.naam, a.straat, a.postnr, a.gemeente*
FROM *tblKlanten as a* ***INNER JOIN*** *artemis.`orders juni 2025`as b*
on a.Klantnummer = b.klantnummer;

Je bekomt hetzelfde resultaat door een query te maken op basis van de tabel tblKlanten en als criteria voor de kolom Klantnummer het sleutelwoord **IN** te typen gevolgd door de SQL-instructie van de eerste query. De SQL-instructie moet tussen ronde haakjes ( ) staan.

De bijhorende SQL-instructie is:

SELECT *Klantnummer, Naam, Straat, Postnr, Gemeente*
FROM *tblKlanten*
WHERE *(Klantnummer IN*

```
        (SELECT DISTINCT Klantnummer
         FROM tblOrders
         WHERE (MONTH(Orderdatum) = 6) AND (YEAR(Orderdatum) = 2005)));
```

Men noemt de binnenste query in dit geval een subquery. Een subquery is een SELECT-instructie die genest is binnen een andere SELECT-instructie, meestal de WHERE- of de HAVING-clausule van een ander SELECT-commando.

De subquery wordt eerst geëvalueerd. Het resultaat wordt dan gebruikt om de hoofd-query te evalueren.

Voorbeeld 2

Maak een query die een antwoord geeft op de vraag: hoeveel klanten wonen er in de gemeenten waarin ook een werknemer woont?

De volgende SQL-instructie geeft het antwoord:

```
SELECT gemeente, COUNT(klantnummer) AS aantal_klanten
FROM tblKlanten
GROUP BY gemeente
HAVING (gemeente IN
        (SELECT DISTINCT gemeente
         FROM tblWerknemers));
```

1.2 Subquery die in één waarde resulteert

Voorbeeld 1

Je wilt de gegevens van de klant die order 10022 geplaatst heeft.

```
SELECT tblKlanten.*
FROM tblKlanten
WHERE (Klantnummer =
        (SELECT Klantnummer
         FROM tblOrders
         WHERE (OrderID = 10022)));
```

Let op: indien de WHERE-component van de hoofdquery een gelijkheidsteken (=) bevat, treedt er een fout op indien de subquery in meer dan één waarde resulteert!

Voorbeeld 2

Je wilt de namen van de klanten kennen waarvan het saldo groter is dan het gemiddelde.

```
SELECT Naam
FROM tblKlanten
WHERE (Saldo >
        (SELECT AVG(Saldo) AS Saldogemiddelde
         FROM tblKlanten AS tblKlanten_1));
```

1.3 Subquery die resulteert in meerdere waarden

Als de WHERE-component van de hoofdquery de SQL-operator IN, ANY of ALL bevat dan mag een subquery resulteren in meerdere waarden.

Voorbeeld 1

Geef de namen van alle klanten die een order geplaatst hebben.

```
SELECT Naam
FROM tblKlanten
WHERE (Klantnummer IN
      (SELECT Klantnummer
       FROM tblOrders));
```

Voorbeeld 2

Selecteer de gegevens van de klanten waarvan het saldo groter is dan het saldo van om het even welke particuliere klant.

```
SELECT tblKlanten.*
FROM tblKlanten
WHERE (Saldo > ALL
      (SELECT Saldo
       FROM tblKlanten AS tblKlanten_1
       WHERE (Type = 'P')));
```

Je gebruikt **ALL** als de relatie moet bestaan t.o.v. alle waarden van de subquery.

Voorbeeld 3

Selecteer de gegevens van de klanten waarvan het saldo groter is dan het saldo van ten minste één particuliere klant.

```
SELECT tblKlanten.*
FROM tblKlanten
WHERE (Saldo > ANY
      (SELECT Saldo
       FROM tblKlanten AS tblKlanten_1
       WHERE (Type = 'P')));
```

Je gebruikt **ANY** als de relatie moet bestaan met ten minste één van de waarden van de subquery.