

# Datenbanken

## Teil 2: Informationen

### Kapitel 10: SQL



## SQL-Unterabfragen

In dieser Einleitung werden die folgenden Unterabfragen vorgestellt:

1. Unterabfragen mit einem Rückgabewert
2. Variablen
3. Unterabfragen mit mehreren Rückgabewerten
4. Beispiel IN -Abfrage
5. Beispiel ANY -Abfrage
6. Beispiel ALL -Abfrage
7. Beispiel EXISTS -Abfrage

## Unterabfragen mit einem Rückgabewert

Unterabfragen, die genau einen Rückgabewert liefern, können direkt mit einem der Vergleichsoperatoren

=	gleich,
<	kleiner,
>	größer,
<>	ungleich,
<=	kleiner oder gleich
>=	größer oder gleich

in eine übergeordnete Abfrage integriert werden.

Nenne den Namen des Mitarbeiters mit dem höchsten Gehalt.

## Unterabfragen mit einem Rückgabewert

Unterabfragen, die genau einen Rückgabewert liefern, können direkt mit einem der Vergleichsoperatoren

=	gleich,
<	kleiner,
>	größer,
<>	ungleich,
<=	kleiner oder gleich
>=	größer oder gleich

in eine übergeordnete Abfrage integriert werden.

Nenne den Namen des Mitarbeiters mit dem höchsten Gehalt.

```
SELECT    MAX (Gehalt)
FROM      Mitarbeiter
```

## Unterabfragen mit einem Rückgabewert

Unterabfragen, die genau einen Rückgabewert liefern, können direkt mit einem der Vergleichsoperatoren

=	gleich,
<	kleiner,
>	größer,
<>	ungleich,
<=	kleiner oder gleich
>=	größer oder gleich

in eine übergeordnete Abfrage integriert werden.

Nenne den Namen des Mitarbeiters mit dem höchsten Gehalt.

```
SELECT      Name
              SELECT      MAX (Gehalt)
              FROM        Mitarbeiter
```

## Unterabfragen mit einem Rückgabewert

Unterabfragen, die genau einen Rückgabewert liefern, können direkt mit einem der Vergleichsoperatoren

=	gleich,
<	kleiner,
>	größer,
<>	ungleich,
<=	kleiner oder gleich
>=	größer oder gleich

in eine übergeordnete Abfrage integriert werden.

Nenne den Namen des Mitarbeiters mit dem höchsten Gehalt.

```
SELECT    Name
FROM      Mitarbeiter
```

```
SELECT    MAX (Gehalt)
FROM      Mitarbeiter
```

## Unterabfragen mit einem Rückgabewert

Unterabfragen, die genau einen Rückgabewert liefern, können direkt mit einem der Vergleichsoperatoren

= gleich,  
< kleiner,  
> größer,  
<> ungleich,  
<= kleiner oder gleich  
>= größer oder gleich

in eine übergeordnete Abfrage integriert werden.

Nenne den Namen des Mitarbeiters mit dem höchsten Gehalt.

```
SELECT      Name
FROM        Mitarbeiter
WHERE       Gehalt =
              (
                SELECT    MAX (Gehalt)
                FROM      Mitarbeiter
              ) ;
```

## Variablen

Das Ergebnis von Unterabfragen, die genau einen Rückgabewert liefern, kann auch in einer Variablen gespeichert werden, auf die in einer späteren Abfrage Bezug genommen werden kann.

Nenne den Namen des Mitarbeiters mit dem höchsten Gehalt.

oder

Nenne den Namen des Mitarbeiters mit dem höchsten Gehalt.

## Variablen

Das Ergebnis von Unterabfragen, die genau einen Rückgabewert liefern, kann auch in einer Variablen gespeichert werden, auf die in einer späteren Abfrage Bezug genommen werden kann.

Nenne den Namen des Mitarbeiters mit dem höchsten Gehalt.

```
SELECT    Name
FROM      Mitarbeiter
WHERE      Gehalt = @gehalt;
```

oder

Nenne den Namen des Mitarbeiters mit dem höchsten Gehalt.

```
SELECT    Name
FROM      Mitarbeiter
WHERE      Gehalt = @gehalt;
```



## Variablen

Das Ergebnis von Unterabfragen, die genau einen Rückgabewert liefern, kann auch in einer Variablen gespeichert werden, auf die in einer späteren Abfrage Bezug genommen werden kann.

Nenne den Namen des Mitarbeiters mit dem höchsten Gehalt.

```
SET          @gehalt =  
  
SELECT      Name  
FROM        Mitarbeiter  
WHERE       Gehalt = @gehalt;
```

oder

Nenne den Namen des Mitarbeiters mit dem höchsten Gehalt.

```
SELECT      Name  
FROM        Mitarbeiter  
WHERE       Gehalt = @gehalt;
```

## Variablen

Das Ergebnis von Unterabfragen, die genau einen Rückgabewert liefern, kann auch in einer Variablen gespeichert werden, auf die in einer späteren Abfrage Bezug genommen werden kann.

Nenne den Namen des Mitarbeiters mit dem höchsten Gehalt.

```
SET      @gehalt =  
          (  
    SELECT  MAX (Gehalt)  
    FROM    Mitarbeiter      ) ;
```

```
SELECT   Name  
FROM     Mitarbeiter  
WHERE    Gehalt = @gehalt;
```

oder

Nenne den Namen des Mitarbeiters mit dem höchsten Gehalt.

```
SELECT   Name  
FROM     Mitarbeiter  
WHERE    Gehalt = @gehalt;
```

## Variablen

Das Ergebnis von Unterabfragen, die genau einen Rückgabewert liefern, kann auch in einer Variablen gespeichert werden, auf die in einer späteren Abfrage Bezug genommen werden kann.

Nenne den Namen des Mitarbeiters mit dem höchsten Gehalt.

```
SET      @gehalt =  
          (  
    SELECT  MAX (Gehalt)  
    FROM    Mitarbeiter      ) ;
```

```
SELECT   Name  
FROM     Mitarbeiter  
WHERE    Gehalt = @gehalt;
```

oder

Nenne den Namen des Mitarbeiters mit dem höchsten Gehalt.

```
SELECT   @gehalt = MAX (Gehalt)
```

```
SELECT   Name  
FROM     Mitarbeiter  
WHERE    Gehalt = @gehalt;
```

## Variablen

Das Ergebnis von Unterabfragen, die genau einen Rückgabewert liefern, kann auch in einer Variablen gespeichert werden, auf die in einer späteren Abfrage Bezug genommen werden kann.

Nenne den Namen des Mitarbeiters mit dem höchsten Gehalt.

```
SET      @gehalt =  
          (  
    SELECT  MAX (Gehalt)  
    FROM    Mitarbeiter      ) ;
```

```
SELECT   Name  
FROM     Mitarbeiter  
WHERE    Gehalt = @gehalt;
```

oder

Nenne den Namen des Mitarbeiters mit dem höchsten Gehalt.

```
SELECT   @gehalt = MAX (Gehalt)  
FROM     Mitarbeiter;
```

```
SELECT   Name  
FROM     Mitarbeiter  
WHERE    Gehalt = @gehalt;
```

## Unterabfragen mit mehreren Rückgabewerten

Unterabfragen, die mehr als einen Rückgabewert zurückliefern, können mit Hilfe der Schlüsselwörter IN, ANY, ALL und EXISTS in eine übergeordnete Abfrage integriert werden. Dabei haben die Schlüsselwörter folgende Bedeutung:

## Unterabfragen mit mehreren Rückgabewerten

Unterabfragen, die mehr als einen Rückgabewert zurückliefern, können mit Hilfe der Schlüsselwörter IN, ANY, ALL und EXISTS in eine übergeordnete Abfrage integriert werden. Dabei haben die Schlüsselwörter folgende Bedeutung:

<b>IN</b>	Wert muss in der Ergebnismenge der Unterabfrage enthalten sein.

## Unterabfragen mit mehreren Rückgabewerten

Unterabfragen, die mehr als einen Rückgabewert zurückliefern, können mit Hilfe der Schlüsselwörter IN, ANY, ALL und EXISTS in eine übergeordnete Abfrage integriert werden. Dabei haben die Schlüsselwörter folgende Bedeutung:

<b>IN</b>	Wert muss in der Ergebnismenge der Unterabfrage enthalten sein.
	Kann mit <b>NOT</b> negiert werden

## Unterabfragen mit mehreren Rückgabewerten

Unterabfragen, die mehr als einen Rückgabewert zurückliefern, können mit Hilfe der Schlüsselwörter IN, ANY, ALL und EXISTS in eine übergeordnete Abfrage integriert werden. Dabei haben die Schlüsselwörter folgende Bedeutung:

<b>IN</b>	Wert muss in der Ergebnismenge der Unterabfrage enthalten sein.
	Kann mit <b>NOT</b> negiert werden
<b>ANY</b> <b>SOME</b>	Bedingung muss für <b>irgendeinen</b> Wert der Unterabfrage gelten.



## Unterabfragen mit mehreren Rückgabewerten

Unterabfragen, die mehr als einen Rückgabewert zurückliefern, können mit Hilfe der Schlüsselwörter IN, ANY, ALL und EXISTS in eine übergeordnete Abfrage integriert werden. Dabei haben die Schlüsselwörter folgende Bedeutung:

<b>IN</b>	Wert muss in der Ergebnismenge der Unterabfrage enthalten sein.
	Kann mit <b>NOT</b> negiert werden
<b>ANY</b> <b>SOME</b>	Bedingung muss für <b>irgendeinen</b> Wert der Unterabfrage gelten.
	Wird mit Vergleichsoperator verwendet: <b>=, &lt;, &gt;, &lt;&gt;, &lt;=, &gt;=</b>

## Unterabfragen mit mehreren Rückgabewerten

Unterabfragen, die mehr als einen Rückgabewert zurückliefern, können mit Hilfe der Schlüsselwörter IN, ANY, ALL und EXISTS in eine übergeordnete Abfrage integriert werden. Dabei haben die Schlüsselwörter folgende Bedeutung:

<b>IN</b>	Wert muss in der Ergebnismenge der Unterabfrage enthalten sein.
	Kann mit <b>NOT</b> negiert werden
<b>ANY</b> <b>SOME</b>	Bedingung muss für <b>irgendeinen</b> Wert der Unterabfrage gelten.
	Wird mit Vergleichsoperator verwendet: <b>=, &lt;, &gt;, &lt;&gt;, &lt;=, &gt;=</b>
	<b>= ANY</b> entspricht <b>IN</b>

## Unterabfragen mit mehreren Rückgabewerten

Unterabfragen, die mehr als einen Rückgabewert zurückliefern, können mit Hilfe der Schlüsselwörter IN, ANY, ALL und EXISTS in eine übergeordnete Abfrage integriert werden. Dabei haben die Schlüsselwörter folgende Bedeutung:

<b>IN</b>	Wert muss in der Ergebnismenge der Unterabfrage enthalten sein.
	Kann mit <b>NOT</b> negiert werden
<b>ANY</b> <b>SOME</b>	Bedingung muss für <b>irgendeinen</b> Wert der Unterabfrage gelten.
	Wird mit Vergleichsoperator verwendet: <b>=, &lt;, &gt;, &lt;&gt;, &lt;=, &gt;=</b>
	<b>= ANY</b> entspricht <b>IN</b>
<b>ALL</b>	Bedingung muss für <b>alle</b> Wert der Unterabfrage gelten.

## Unterabfragen mit mehreren Rückgabewerten

Unterabfragen, die mehr als einen Rückgabewert zurückliefern, können mit Hilfe der Schlüsselwörter IN, ANY, ALL und EXISTS in eine übergeordnete Abfrage integriert werden. Dabei haben die Schlüsselwörter folgende Bedeutung:

<b>IN</b>	Wert muss in der Ergebnismenge der Unterabfrage enthalten sein.
	Kann mit <b>NOT</b> negiert werden
<b>ANY</b> <b>SOME</b>	Bedingung muss für <b>irgendeinen</b> Wert der Unterabfrage gelten.
	Wird mit Vergleichsoperator verwendet: <b>=, &lt;, &gt;, &lt;&gt;, &lt;=, &gt;=</b>
	<b>= ANY</b> entspricht <b>IN</b>
<b>ALL</b>	Bedingung muss für <b>alle</b> Wert der Unterabfrage gelten.
	Wird mit Vergleichsoperator verwendet: <b>=, &lt;, &gt;, &lt;&gt;, &lt;=, &gt;=</b>

## Unterabfragen mit mehreren Rückgabewerten

Unterabfragen, die mehr als einen Rückgabewert zurückliefern, können mit Hilfe der Schlüsselwörter IN, ANY, ALL und EXISTS in eine übergeordnete Abfrage integriert werden. Dabei haben die Schlüsselwörter folgende Bedeutung:

<b>IN</b>	Wert muss in der Ergebnismenge der Unterabfrage enthalten sein.
	Kann mit <b>NOT</b> negiert werden
<b>ANY</b> <b>SOME</b>	Bedingung muss für <b>irgendeinen</b> Wert der Unterabfrage gelten.
	Wird mit Vergleichsoperator verwendet: <b>=, &lt;, &gt;, &lt;&gt;, &lt;=, &gt;=</b>
	<b>= ANY</b> entspricht <b>IN</b>
<b>ALL</b>	Bedingung muss für <b>alle</b> Wert der Unterabfrage gelten.
	Wird mit Vergleichsoperator verwendet: <b>=, &lt;, &gt;, &lt;&gt;, &lt;=, &gt;=</b>
	<b>&lt;&gt; ALL</b> entspricht <b>NOT IN</b>

## Unterabfragen mit mehreren Rückgabewerten

Unterabfragen, die mehr als einen Rückgabewert zurückliefern, können mit Hilfe der Schlüsselwörter IN, ANY, ALL und EXISTS in eine übergeordnete Abfrage integriert werden. Dabei haben die Schlüsselwörter folgende Bedeutung:

<b>IN</b>	Wert muss in der Ergebnismenge der Unterabfrage enthalten sein.
	Kann mit <b>NOT</b> negiert werden
<b>ANY</b> <b>SOME</b>	Bedingung muss für <b>irgendeinen</b> Wert der Unterabfrage gelten.
	Wird mit Vergleichsoperator verwendet: <b>=, &lt;, &gt;, &lt;&gt;, &lt;=, &gt;=</b>
	<b>= ANY</b> entspricht <b>IN</b>
<b>ALL</b>	Bedingung muss für <b>alle</b> Wert der Unterabfrage gelten.
	Wird mit Vergleichsoperator verwendet: <b>=, &lt;, &gt;, &lt;&gt;, &lt;=, &gt;=</b>
	<b>&lt;&gt; ALL</b> entspricht <b>NOT IN</b>
<b>EXISTS</b>	Existenz von Ergebnisdatensätzen einer Unterabfrage wird geprüft

## Unterabfragen mit mehreren Rückgabewerten

Unterabfragen, die mehr als einen Rückgabewert zurückliefern, können mit Hilfe der Schlüsselwörter IN, ANY, ALL und EXISTS in eine übergeordnete Abfrage integriert werden. Dabei haben die Schlüsselwörter folgende Bedeutung:

<b>IN</b>	Wert muss in der Ergebnismenge der Unterabfrage enthalten sein.
	Kann mit <b>NOT</b> negiert werden
<b>ANY</b> <b>SOME</b>	Bedingung muss für <b>irgendeinen</b> Wert der Unterabfrage gelten.
	Wird mit Vergleichsoperator verwendet: =, <, >, <>, <=, >=
	= <b>ANY</b> entspricht <b>IN</b>
<b>ALL</b>	Bedingung muss für <b>alle</b> Wert der Unterabfrage gelten.
	Wird mit Vergleichsoperator verwendet: =, <, >, <>, <=, >=
	<> <b>ALL</b> entspricht <b>NOT IN</b>
<b>EXISTS</b>	Existenz von Ergebnisdatensätzen einer Unterabfrage wird geprüft
	Kann mit <b>NOT</b> negiert werden

## Unterabfragen mit mehreren Rückgabewerten

### *Mitarbeiter*

<u>ID_Mitarbeiter</u>	Name	Gehalt	ID_Abteilung
1	Specht	2000	1
2	Ernst	2000	2
3	Müller	2700	1
4	Schulte	3000	2

### *Abteilung*

<u>ID_Abteilung</u>	Bezeichnung
1	Entwicklung
2	Fertigung



# Unterabfragen mit mehreren Rückgabewerten

## Mitarbeiter

<u>ID_Mitarbeiter</u>	Name	Gehalt	ID_Abteilung
1	Specht	2000	1
2	Ernst	2000	2
3	Müller	2700	1
4	Schulte	3000	2

## Abteilung

<u>ID_Abteilung</u>	Bezeichnung
1	Entwicklung
2	Fertigung

## SQL-Abfrage

Nenne die Namen aller Mitarbeiter, deren Gehalt in selber Höhe auch an einen Mitarbeiter aus der Abteilung "Entwicklung" gezahlt wird.

# Unterabfragen mit mehreren Rückgabewerten

## Mitarbeiter

ID_Mitarbeiter	Name	Gehalt	ID_Abteilung
1	Specht	2000	1
2	Ernst	2000	2
3	Müller	2700	1
4	Schulte	3000	2

## Abteilung

ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung

## SQL-Abfrage

Nenne die Namen aller Mitarbeiter, deren Gehalt in selber Höhe auch an einen Mitarbeiter aus der Abteilung "Entwicklung" gezahlt wird.

```
SELECT    M.Gehalt
FROM      Mitarbeiter M, Abteilung A
WHERE     A.ID_Abteilung = M.ID_Abteilung
AND      A.Bezeichnung = "Entwicklung"
```

# Unterabfragen mit mehreren Rückgabewerten

## Mitarbeiter

ID_Mitarbeiter	Name	Gehalt	ID_Abteilung
1	Specht	2000	1
2	Ernst	2000	2
3	Müller	2700	1
4	Schulte	3000	2

## Abteilung

ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung

## SQL-Abfrage

Nenne die Namen aller Mitarbeiter, deren Gehalt in selber Höhe auch an einen Mitarbeiter aus der Abteilung "Entwicklung" gezahlt wird.

```

SELECT    M.Gehalt
FROM      Mitarbeiter M, Abteilung A
WHERE     A.ID_Abteilung = M.ID_Abteilung
AND      A.Bezeichnung = "Entwicklung"
  
```

Ausgabe:  
2000  
2700

# Unterabfragen mit mehreren Rückgabewerten

## Mitarbeiter

ID_Mitarbeiter	Name	Gehalt	ID_Abteilung
1	Specht	2000	1
2	Ernst	2000	2
3	Müller	2700	1
4	Schulte	3000	2

## Abteilung

ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung

## SQL-Abfrage

Nenne die Namen aller Mitarbeiter, deren Gehalt in selber Höhe auch an einen Mitarbeiter aus der Abteilung "Entwicklung" gezahlt wird.

```

SELECT      Name
FROM        Mitarbeiter
WHERE       Gehalt IN
            (
SELECT      M.Gehalt
FROM        Mitarbeiter M, Abteilung A
WHERE       A.ID_Abteilung = M.ID_Abteilung
AND         A.Bezeichnung = "Entwicklung"
            );

```

Ausgabe:  
2000  
2700

# Unterabfragen mit mehreren Rückgabewerten

## Mitarbeiter

ID_Mitarbeiter	Name	Gehalt	ID_Abteilung
1	Specht	2000	1
2	Ernst	2000	2
3	Müller	2700	1
4	Schulte	3000	2

## Abteilung

ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung

## SQL-Abfrage

Nenne die Namen aller Mitarbeiter, deren Gehalt in selber Höhe auch an einen Mitarbeiter aus der Abteilung "Entwicklung" gezahlt wird.

```

SELECT      Name
FROM        Mitarbeiter
WHERE       Gehalt IN
              (
                SELECT    M.Gehalt
                FROM      Mitarbeiter M, Abteilung A
                WHERE     A.ID_Abteilung = M.ID_Abteilung
                AND       A.Bezeichnung = "Entwicklung"      );
  
```

Ausgabe:  
2000  
2700

Ausgabe: Ernst, Specht, Müller

## Unterabfragen mit mehreren Rückgabewerten

### *Mitarbeiter*

<u>ID_Mitarbeiter</u>	Name	Gehalt	ID_Abteilung
1	Specht	2000	1
2	Ernst	2000	2
3	Müller	2700	1
4	Schulte	3000	2

### *Abteilung*

<u>ID_Abteilung</u>	Bezeichnung
1	Entwicklung
2	Fertigung

## Unterabfragen mit mehreren Rückgabewerten

### Mitarbeiter

<u>ID_Mitarbeiter</u>	Name	Gehalt	ID_Abteilung
1	Specht	2000	1
2	Ernst	2000	2
3	Müller	2700	1
4	Schulte	3000	2

### Abteilung

<u>ID_Abteilung</u>	Bezeichnung
1	Entwicklung
2	Fertigung

### SQL-Abfrage

Nenne die Namen aller Mitarbeiter, deren Gehalt niedriger ist als irgendein Gehalt von Mitarbeitern aus der Abteilung "Entwicklung".

# Unterabfragen mit mehreren Rückgabewerten

## Mitarbeiter

ID_Mitarbeiter	Name	Gehalt	ID_Abteilung
1	Specht	2000	1
2	Ernst	2000	2
3	Müller	2700	1
4	Schulte	3000	2

## Abteilung

ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung

## SQL-Abfrage

Nenne die Namen aller Mitarbeiter, deren Gehalt niedriger ist als irgendein Gehalt von Mitarbeitern aus der Abteilung "Entwicklung".

```
SELECT M.Gehalt
FROM Mitarbeiter M, Abteilung A
WHERE A.ID_Abteilung = M.ID_Abteilung
AND A.Bezeichnung = "Entwicklung"
```



# Unterabfragen mit mehreren Rückgabewerten

## Mitarbeiter

ID_Mitarbeiter	Name	Gehalt	ID_Abteilung
1	Specht	2000	1
2	Ernst	2000	2
3	Müller	2700	1
4	Schulte	3000	2

## Abteilung

ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung

## SQL-Abfrage

Nenne die Namen aller Mitarbeiter, deren Gehalt niedriger ist als irgendein Gehalt von Mitarbeitern aus der Abteilung "Entwicklung".

```

SELECT    M.Gehalt
FROM      Mitarbeiter M, Abteilung A
WHERE     A.ID_Abteilung = M.ID_Abteilung
AND      A.Bezeichnung = "Entwicklung"
  
```

Ausgabe:  
2000  
2700

# Unterabfragen mit mehreren Rückgabewerten

## Mitarbeiter

ID_Mitarbeiter	Name	Gehalt	ID_Abteilung
1	Specht	2000	1
2	Ernst	2000	2
3	Müller	2700	1
4	Schulte	3000	2

## Abteilung

ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung

## SQL-Abfrage

Nenne die Namen aller Mitarbeiter, deren Gehalt niedriger ist als irgendein Gehalt von Mitarbeitern aus der Abteilung "Entwicklung".

```

SELECT      Name
FROM        Mitarbeiter
WHERE       Gehalt < ANY
            (
              SELECT      M.Gehalt
              FROM        Mitarbeiter M, Abteilung A
              WHERE       A.ID_Abteilung = M.ID_Abteilung
              AND         A.Bezeichnung = "Entwicklung"
            );
  
```

Ausgabe:  
2000  
2700

# Unterabfragen mit mehreren Rückgabewerten

## Mitarbeiter

ID_Mitarbeiter	Name	Gehalt	ID_Abteilung
1	Specht	2000	1
2	Ernst	2000	2
3	Müller	2700	1
4	Schulte	3000	2

## Abteilung

ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung

## SQL-Abfrage

Nenne die Namen aller Mitarbeiter, deren Gehalt niedriger ist als irgendein Gehalt von Mitarbeitern aus der Abteilung "Entwicklung".

```

SELECT      Name
FROM        Mitarbeiter
WHERE       Gehalt < ANY
              (
                SELECT  M.Gehalt
                FROM    Mitarbeiter M, Abteilung A
                WHERE   A.ID_Abteilung = M.ID_Abteilung
                AND     A.Bezeichnung = "Entwicklung"
              );
  
```

*Ausgabe:*  
2000  
2700

*Ausgabe:* Es gibt keinen Mitarbeiter mit Gehalt < 2000

## Unterabfragen mit mehreren Rückgabewerten

### *Mitarbeiter*

<u>ID_Mitarbeiter</u>	Name	Gehalt	ID_Abteilung
1	Specht	2000	1
2	Ernst	2000	2
3	Müller	2700	1
4	Schulte	3000	2

### *Abteilung*

<u>ID_Abteilung</u>	Bezeichnung
1	Entwicklung
2	Fertigung

# Unterabfragen mit mehreren Rückgabewerten

## Mitarbeiter

<u>ID_Mitarbeiter</u>	Name	Gehalt	ID_Abteilung
1	Specht	2000	1
2	Ernst	2000	2
3	Müller	2700	1
4	Schulte	3000	2

## Abteilung

<u>ID_Abteilung</u>	Bezeichnung
1	Entwicklung
2	Fertigung

## SQL-Abfrage

Nenne die Namen aller Mitarbeiter, deren Gehalt höher ist als alle Gehalt von Mitarbeitern aus der Abteilung "Entwicklung".

# Unterabfragen mit mehreren Rückgabewerten

## Mitarbeiter

ID_Mitarbeiter	Name	Gehalt	ID_Abteilung
1	Specht	2000	1
2	Ernst	2000	2
3	Müller	2700	1
4	Schulte	3000	2

## Abteilung

ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung

## SQL-Abfrage

Nenne die Namen aller Mitarbeiter, deren Gehalt höher ist als alle Gehalt von Mitarbeitern aus der Abteilung "Entwicklung".

```
SELECT M.Gehalt
FROM Mitarbeiter M, Abteilung A
WHERE A.ID_Abteilung = M.ID_Abteilung
AND A.Bezeichnung = "Entwicklung" );
```

# Unterabfragen mit mehreren Rückgabewerten

## Mitarbeiter

ID_Mitarbeiter	Name	Gehalt	ID_Abteilung
1	Specht	2000	1
2	Ernst	2000	2
3	Müller	2700	1
4	Schulte	3000	2

## Abteilung

ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung

## SQL-Abfrage

Nenne die Namen aller Mitarbeiter, deren Gehalt höher ist als alle Gehalt von Mitarbeitern aus der Abteilung "Entwicklung".

```
SELECT M.Gehalt
FROM Mitarbeiter M, Abteilung A
WHERE A.ID_Abteilung = M.ID_Abteilung
AND A.Bezeichnung = "Entwicklung"
```

Ausgabe:  
2000  
2700

# Unterabfragen mit mehreren Rückgabewerten

## Mitarbeiter

ID_Mitarbeiter	Name	Gehalt	ID_Abteilung
1	Specht	2000	1
2	Ernst	2000	2
3	Müller	2700	1
4	Schulte	3000	2

## Abteilung

ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung

## SQL-Abfrage

Nenne die Namen aller Mitarbeiter, deren Gehalt höher ist als alle Gehalt von Mitarbeitern aus der Abteilung "Entwicklung".

```

SELECT      Name
FROM        Mitarbeiter
WHERE       Gehalt > ALL
            (
SELECT      M.Gehalt
FROM        Mitarbeiter M, Abteilung A
WHERE       A.ID_Abteilung = M.ID_Abteilung
AND         A.Bezeichnung = "Entwicklung"
            );

```

Ausgabe:  
2000  
2700



# Unterabfragen mit mehreren Rückgabewerten

## Mitarbeiter

ID_Mitarbeiter	Name	Gehalt	ID_Abteilung
1	Specht	2000	1
2	Ernst	2000	2
3	Müller	2700	1
4	Schulte	3000	2

## Abteilung

ID_Abteilung	Bezeichnung
1	Entwicklung
2	Fertigung

## SQL-Abfrage

Nenne die Namen aller Mitarbeiter, deren Gehalt höher ist als alle Gehalt von Mitarbeitern aus der Abteilung "Entwicklung".

```

SELECT      Name
FROM        Mitarbeiter
WHERE       Gehalt > ALL
              (
SELECT      M.Gehalt
FROM        Mitarbeiter M, Abteilung A
WHERE       A.ID_Abteilung = M.ID_Abteilung
AND        A.Bezeichnung = "Entwicklung"  );
  
```

*Ausgabe:*  
2000  
2700

*Ausgabe:* Schulte

## Unterabfragen mit mehreren Rückgabewerten

### *Mitarbeiter*

<u>ID_Mitarbeiter</u>	Nachname	Vorname
1	Specht	Willi
2	Ernst	Adam
3	Müller	Silke
4	Schulte	Eva

### *Kunde*

<u>ID_Kunde</u>	Nachname	Vorname
1	Klein	Beate
2	Ernst	Silke
3	Schulte	Eva

## Unterabfragen mit mehreren Rückgabewerten

### Mitarbeiter

<u>ID_Mitarbeiter</u>	Nachname	Vorname
1	Specht	Willi
2	Ernst	Adam
3	Müller	Silke
4	Schulte	Eva

### Kunde

<u>ID_Kunde</u>	Nachname	Vorname
1	Klein	Beate
2	Ernst	Silke
3	Schulte	Eva

### SQL-Abfrage

Nenne die Vor- und Nachnamen aller Mitarbeiter, deren Vor- und Nachnamenskombination auch in der Kundentabelle vorkommt.

## Unterabfragen mit mehreren Rückgabewerten

### Mitarbeiter

<u>ID_Mitarbeiter</u>	Nachname	Vorname
1	Specht	Willi
2	Ernst	Adam
3	Müller	Silke
4	Schulte	Eva

### Kunde

<u>ID_Kunde</u>	Nachname	Vorname
1	Klein	Beate
2	Ernst	Silke
3	Schulte	Eva

### SQL-Abfrage

Nenne die Vor- und Nachnamen aller Mitarbeiter, deren Vor- und Nachnamenskombination auch in der Kundentabelle vorkommt.

```
SELECT *  
FROM Kunde K
```

# Unterabfragen mit mehreren Rückgabewerten

## Mitarbeiter

<u>ID_Mitarbeiter</u>	Nachname	Vorname
1	Specht	Willi
2	Ernst	Adam
3	Müller	Silke
4	Schulte	Eva

## Kunde

<u>ID_Kunde</u>	Nachname	Vorname
1	Klein	Beate
2	Ernst	Silke
3	Schulte	Eva

## SQL-Abfrage

Nenne die Vor- und Nachnamen aller Mitarbeiter, deren Vor- und Nachnamenskombination auch in der Kundentabelle vorkommt.

```
FROM      Mitarbeiter M  
  
      SELECT  *  
      FROM      Kunde K
```

## Unterabfragen mit mehreren Rückgabewerten

### Mitarbeiter

<u>ID_Mitarbeiter</u>	Nachname	Vorname
1	Specht	Willi
2	Ernst	Adam
3	Müller	Silke
4	Schulte	Eva

### Kunde

<u>ID_Kunde</u>	Nachname	Vorname
1	Klein	Beate
2	Ernst	Silke
3	Schulte	Eva

### SQL-Abfrage

Nenne die Vor- und Nachnamen aller Mitarbeiter, deren Vor- und Nachnamenskombination auch in der Kundentabelle vorkommt.

```

FROM      Mitarbeiter M

      (
        SELECT   *
        FROM     Kunde K
        WHERE    M.Vorname = K.Vorname
        AND     M.Nachname = K.Nachname      );
  
```

## Unterabfragen mit mehreren Rückgabewerten

### Mitarbeiter

ID_Mitarbeiter	Nachname	Vorname
1	Specht	Willi
2	Ernst	Adam
3	Müller	Silke
4	Schulte	Eva

### Kunde

ID_Kunde	Nachname	Vorname
1	Klein	Beate
2	Ernst	Silke
3	Schulte	Eva

### SQL-Abfrage

Nenne die Vor- und Nachnamen aller Mitarbeiter, deren Vor- und Nachnamenskombination auch in der Kundentabelle vorkommt.

```

FROM      Mitarbeiter M
(
  SELECT   *
  FROM     Kunde K
  WHERE    M.Vorname = K.Vorname
  AND     M.Nachname = K.Nachname  );
  
```

*Ausgabe:  
3, Schulte, Eva*

## Unterabfragen mit mehreren Rückgabewerten

### Mitarbeiter

ID_Mitarbeiter	Nachname	Vorname
1	Specht	Willi
2	Ernst	Adam
3	Müller	Silke
4	Schulte	Eva

### Kunde

ID_Kunde	Nachname	Vorname
1	Klein	Beate
2	Ernst	Silke
3	Schulte	Eva

### SQL-Abfrage

Nenne die Vor- und Nachnamen aller Mitarbeiter, deren Vor- und Nachnamenskombination auch in der Kundentabelle vorkommt.

```

SELECT    M.Vorname, M.Nachname
FROM      Mitarbeiter M

(
  SELECT    *
  FROM      Kunde K
  WHERE     M.Vorname = K.Vorname
  AND      M.Nachname = K.Nachname  );
  
```

*Ausgabe:*  
3, Schulte, Eva



## Unterabfragen mit mehreren Rückgabewerten

### Mitarbeiter

ID_Mitarbeiter	Nachname	Vorname
1	Specht	Willi
2	Ernst	Adam
3	Müller	Silke
4	Schulte	Eva

### Kunde

ID_Kunde	Nachname	Vorname
1	Klein	Beate
2	Ernst	Silke
3	Schulte	Eva

### SQL-Abfrage

Nenne die Vor- und Nachnamen aller Mitarbeiter, deren Vor- und Nachnamenskombination auch in der Kundentabelle vorkommt.

```

SELECT      M.Vorname, M.Nachname
FROM        Mitarbeiter M
WHERE        EXISTS
                (
                  SELECT      *
                  FROM        Kunde K
                  WHERE      M.Vorname = K.Vorname
                  AND        M.Nachname = K.Nachname      );
  
```

*Ausgabe:*  
3, Schulte, Eva

## Unterabfragen mit mehreren Rückgabewerten

### Mitarbeiter

ID_Mitarbeiter	Nachname	Vorname
1	Specht	Willi
2	Ernst	Adam
3	Müller	Silke
4	Schulte	Eva

### Kunde

ID_Kunde	Nachname	Vorname
1	Klein	Beate
2	Ernst	Silke
3	Schulte	Eva

### SQL-Abfrage

Nenne die Vor- und Nachnamen aller Mitarbeiter, deren Vor- und Nachnamenskombination auch in der Kundentabelle vorkommt.

```

SELECT      M.Vorname, M.Nachname
FROM        Mitarbeiter M
WHERE        EXISTS
                (
                  SELECT      *
                  FROM        Kunde K
                  WHERE      M.Vorname = K.Vorname
                  AND        M.Nachname = K.Nachname      );

```

*Ausgabe:*  
3, Schulte, Eva

*Ausgabe:* Eva, Schulte