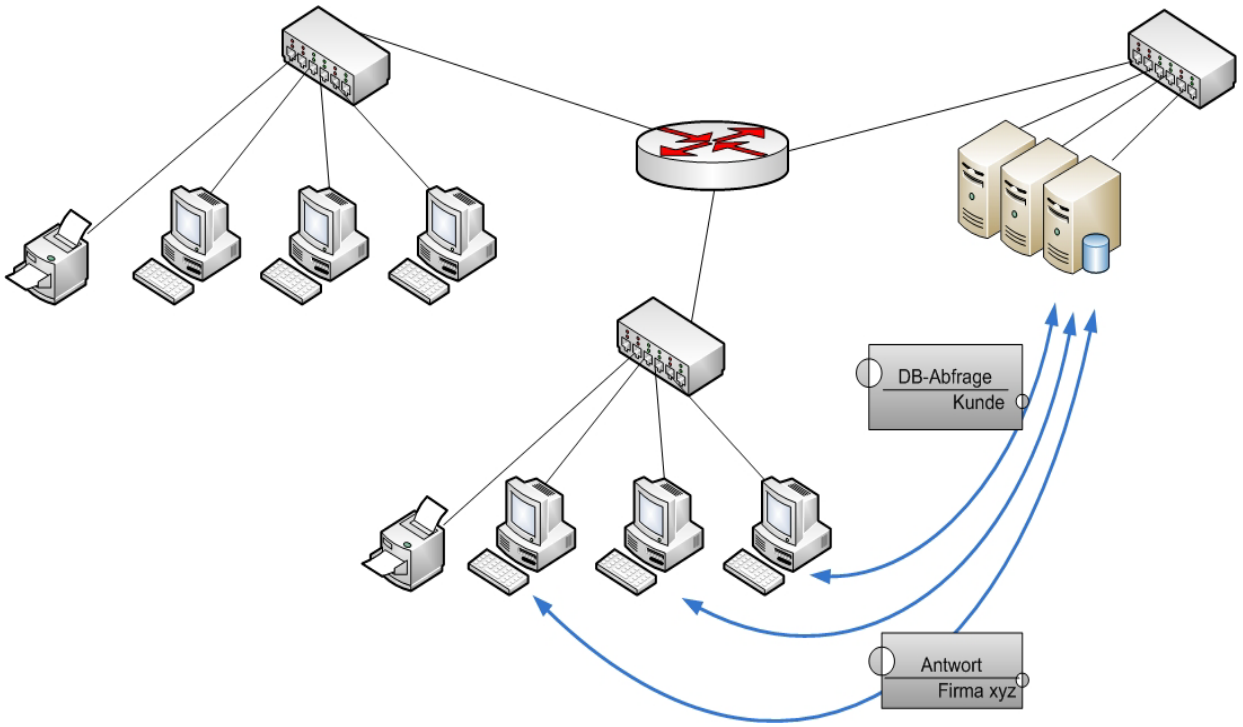
Unternehmens-Datenbanken im Netzwerk

Teil 3: SQL Programmierung

- Grundlagen
- SQL-Befehle
- Funktionen
- Operatoren
- Datentypen
- Datenbank erstellen
- Tabellen
- SQL-Abfragen

Autor: Rainer Egewardt



Kompaktes Datenbank-Wissen rund um die Datenbank-Programmierung mit Transact-SQL

Unsere Bücher „Das PC-Wissen für IT-Berufe“ als Print-Medien, sind zu Bestsellern im IT-Buchmarkt geworden.
Hier nun auch kompaktes IT-Wissen als ebook.

Powered by

PCT Solutions

Inhaltsverzeichnis

Dieses ebook wurde auf der Basis von fundierten Ausbildungen, Weiterbildungen und umfangreichen Praxiserfahrungen erstellt. Für Schäden aus unvollständigen oder fehlerhaften Informationen übernehmen wir jedoch keinerlei Haftung.

PCT-Solutions

info@pct-solutions.de
www.pct-solutions.de

Überblick über die einzelnen Kapitel

Tipp: Für ein detailliertes Inhaltsverzeichnis mit allen Unterpunkten benutzen Sie bitte die Lesezeichen links im AcrobatReader. Hier kann schnell und direkt zu den einzelnen Punkten und Kapiteln gesprungen werden.

IT-Lösungen • Dokumentationen • Präsentationen

Top-aktuelle Ebooks als Download

Unsere top-aktuellen Neuveröffentlichungen als EBooks zum Download von unserer Web-Site

PCT-Solutions

info@pct-solutions.de
www.pct-solutions.de

- Computer-Netzwerke Teil 1	Netzwerk-Design (Netzwerk-Hardware)
- Computer-Netzwerke Teil 2	Konfiguration eines Windows-Server basierten Netzwerkes
- Computer-Netzwerke Teil 3	DNS-, WINS-, DHCP-Konfiguration
- Computer-Netzwerke Teil 4	Optimieren von Windows-Netzwerken
- Computer-Netzwerke Teil 5	Netzwerkanbindung von Windows-Clients
- Computer-Netzwerke Teil 6	Scripting-Host in IT-Netzwerken
- Computer-Netzwerke Teil 7	Projekt-Management in IT-Netzwerken
- Datenbank Teil 1	MS-SQL-Server als Datenbank-Backend
- Datenbank Teil 2	MS-Access als Datenbank-Frontend
- Datenbank Teil 3	SQL-Programmierung (Transact-SQL)
- Mailing Teil 1	MS-Exchange-Server als Mail-Server
- Mailing Teil 2	Outlook als Mail-Client
- Internet Teil 1	Internet-Information-Server als HTML-Server
- Internet Teil 2	MS-Frontpage zum Erstellen eines HTML-Pools
- Internet Teil 3	Internet-Browser
- Web-Programmierung Teil 1	HTML
- Web-Programmierung Teil 1	DHTML
- Web-Programmierung Teil 1	CSS
- Web-Programmierung Teil 1	PHP
- Web-Programmierung Teil 1	JavaScript
- Web-Programmierung Teil 1	XML
- Software Teil 1	Professionelle Bildbearbeitung Corel PhotoPaint
- Software Teil 2	Professionelle Layouts mit Adobe Illustrator
- Software Teil 3	Grafisches Allerlei mit MS-Visio

und viele weitere EBooks zum Download auf unserer Internetseite

Arbeiten mit SQL-Anfragen

Gerüst einer Anfrage
SQL-Anfragen bestehen in erster Linie aus SELECT-Anweisungen. SELECT-Anweisungen wiederum bestehen aus Schlüsselwörtern, den Klauseln. Hierbei hat sich ein Gerüst etabliert, welches aus SELECT, WHERE, FROM, GROUP BY und HAVING besteht.

SELECT
SELECT bestimmt, welche Spalten in der Ergebnismenge vorhanden sein sollen. Dabei können Spalten aus einer oder mehreren Tabellen selektiert werden. Weiterhin sind berechnete Spalten möglich, wobei mathematische Ausdrücke verwendet werden können. SELECT muss in jeder Abfrage enthalten sein.

FROM
FROM legt fest, aus welcher Tabelle oder welchem View Spalten verwendet werden sollen. FROM muss in jeder Abfrage enthalten sein.

WHERE
WHERE filtert Spalten, die mit FROM angegeben wurden. Dabei müssen WHERE Ausdrücke verwendet werden, die auf wahr, falsch, unbekannt geprüft werden können. Diese sind meistens Vergleichsoperatoren, Boolesche Operatoren oder spezielle Operatoren. WHERE muss nicht in jeder Abfrage enthalten sein.

GROUP BY
GROUP BY faßt Informationen zusammen. Hierbei können Daten in Gruppen zerlegt werden. Gruppiert werden kann nach einer oder mehreren Spalten. GROUP BY muss nicht in jeder Abfrage enthalten sein.

HAVING
HAVING wird nur in Kombination mit GROUP BY verwendet. HAVING gruppiert gefilterte Daten. Nach HAVING muss eine Bedingung folgen, mit der auf wahr, falsch, unbekannt geprüft werden kann.

Abfragen erstellen

Eine Abfrage an eine Datenbank zu erstellen bedeutet, dass zuerst ein bestimmtes Problem vorhanden ist, welches über die Abfrage gelöst werden soll. Dazu muss über die Anweisungen das Problem ausgedrückt werden können. Natürlich muss man hierbei wissen, über welche Spalten einer Tabelle das Problem lösbar ist. Ist man sich dessen nicht bewusst, kann auch mit Wildcards gearbeitet werden (*).

Um bei unserem Beispiel zu bleiben: Es gibt angenommen eine Tabelle KUNDEN in der alle Daten der Kunden wie Vorname, Nachname, Ort, usw. in entsprechenden Spalten enthalten sind. Um nun alle Kunden auszuwählen und darzustellen, kann folgende Anweisung verwendet werden:

```
SELECT *
FROM kunden;
```

Doppelte Zeilen vermeiden

Da es immer wieder vorkommen kann, dass doppelte Anzeigen in der Ergebnisanzeige auftreten können, sollte dies mit DISTINCT verhindert werden. Wenn z.B. nur die einfach vorkommenden Nachnamen einer Tabelle ausgegeben werden sollen, kann dies mit folgender Abfrage erledigt werden:

```
SELECT DISTINCT nachname
FROM kunden;
```

DISTINCT kann auch auf mehrere Spalten angewendet werden.

Sortierte Ausgaben

Im Allgemeinen ist die Ausgabe einer Ergebnismenge unsortiert. Soll diese in einer bestimmten Reihenfolge erfolgen, kann dies mit ORDER BY ermöglicht werden. Der Vorgang kann ebenfalls auf eine oder mehrere Spalten angewendet werden, nach denen sortiert werden soll. Aufsteigende Sortierung ist die Vorein-

stellung (ASC). Absteigend kann aber ebenfalls eingestellt werden (DESC):

```
SELECT DISTINCT nachname, vorname
FROM kunden
ORDER BY nachname DESC, vorname
DESC;
```

Berechnete Ausgaben

Spaltenwerte sowie numerische Literale können in einer Ergebnisausgabe mit mathematischen Operatoren verknüpft werden. In der Ausgabe erscheint dann der berechnete Wert. Z.B. sind in einer Tabelle LAGER Spalten über den Einkaufspreis, den Verkaufspreis und der Artikelbezeichnung vorhanden. Um nun die Differenz vom Einkaufs- zum Verkaufspreis, sortiert nach der Artikelbezeichnung zu erhalten, kann folgende Abfrage verwendet werden:

```
SELECT ArtName, EinkPreis – VerkPreis
FROM lager
ORDER BY ArtName;
```

Gefilterte Ausgaben

Sollen nicht alle Zeilen einer Tabelle in einer Ausgabe berücksichtigt werden, wie es in den vorangegangenen Beispielen der Fall war, sondern nur bestimmte Zeilen einer Tabelle, dann muss mit WHERE gearbeitet werden. WHERE arbeitet dabei mit fünf unterschiedlichen Prädikaten. Hierbei wird geprüft, ob ein oder mehrere Wertausdrücke wahr, falsch oder unbekannt ergeben.

Bei einem Vergleich werden zwei Ausdrücke mit den folgenden Operatoren verglichen:
= (gleich), <> (ungleich), > (größer als), < (kleiner als), >= (größer gleich), <= (kleiner gleich)

```
SELECT ArtName, EinkPreis
FROM lager
WHERE EinkPreis = 100
ORDER BY ArtName;
```

Die Operatoren <, >, >=, <= sind auf Buchstaben, Zahlen, Zeichenketten und Datum/Uhrzeit anwendbar.

```
SELECT ArtName, EinkPreis, VerkPreis
FROM lager
WHERE EinkPreis > VerkPreis
ORDER BY ArtName;
```

Mit BETWEEN kann geprüft werden, ob sich ein Wert in einem bestimmten Bereich befindet. Die Wertausdrücke des Bereiches werden durch AND getrennt.

```
SELECT ArtName, VerkPreis
FROM lager
WHERE VerkPreis BETWEEN 100 AND 200
ORDER BY ArtName;
```

Mit IN kann geprüft werden, ob ein Wertausdruck sich in einer bestimmten Menge von Werten befindet.

```
SELECT ArtName, VerkPreis
FROM lager
WHERE ArtName IN ( ' Mainboard ' )
ORDER BY VerkPreis;
```

Mit LIKE können Zeichenketten verglichen werden. Mustervergleiche sind hierbei auch möglich.

```
SELECT ArtName, VerkPreis
FROM lager
WHERE ArtName LIKE ' Mainboard '
ORDER BY VerkPreis;
```

```
SELECT ArtName, VerkPreis
FROM lager
WHERE ArtName LIKE ' Mainb% '
ORDER BY VerkPreis;
```

Mit IS NULL kann geprüft werden, ob ein Wertausdruck einen NULL-Wert ergibt, also kein Eintrag vorhanden ist. Hier ergibt das Ergebnis unbekannt, und nicht wahr oder falsch.

```
SELECT ArtName, VerkPreis
FROM lager
WHERE VerkPreis IS NULL
ORDER BY ArtName;
```

Mit dem Schlüsselwort NOT vor den anderen Schlüsselwörtern können alle obigen Beispiele negiert werden:

NOT BETWEEN
NOT IN
NOT LIKE
IS NOT NULL
usw.

<u>Bedingungen</u>

Die oben besprochenen Abfragen beinhalteten immer nur eine Bedingung. Sollen mehrere Bedingungen verwendet werden, ist dies mit AND und OR möglich.

```
SELECT ArtName, VerkPreis
FROM lager
WHERE VerkPreis > 100
AND ArtName LIKE ' Mainboard '
ORDER BY ArtName;
```

```
SELECT ArtName, VerkPreis
FROM lager
WHERE VerkPreis > 100
OR ArtName LIKE ' Mainboard '
ORDER BY ArtName;
```

Auch Kombinationen von beiden sind möglich.

```
SELECT ArtName, VerkPreis
FROM lager
WHERE VerkPreis > 100
AND ArtName LIKE ' Mainboard '
OR ArtName LIKE ' Festplatte '
ORDER BY ArtName;
```

Aggregatfunktionen

Aggregatfunktionen ermöglichen folgende Abläufe:

- Berechnungen durchführen
- Zeilen oder Werte zählen
- Summen über Spaltenwerte bilden
- Zeilen einer Tabelle zählen
- Mittelwerte über Spaltenwerte bilden
- Den kleinsten oder größten Wert einer Spalte heraus finden

Mittels COUNT können die Zeilen einer Ergebnismenge gezählt werden. Hierbei kann COUNT(*) für das Zählen aller Zeilen, oder COUNT Ausdruck für das Zählen der Zeilenanzahl des Ausdrucks verwendet werden. NULL-Werte werden hierbei mitgezählt.

Die folgende Abfrage zählt alle Artikel in der Tabelle Lager:

```
SELECT COUNT(*)
FROM lager;
```

Um festzustellen, wie viele verschiedene Mainboards im Lager vorhanden sind, kann folgende Abfrage verwendet werden:

```
SELECT COUNT(*)
FROM lager
WHERE ArtName = ' Mainboard ';
```

Mit SUM können Summenberechnungen von Spalten durchgeführt werden. NULL-Werte werden dabei nicht berücksichtigt:

```
SELECT  SUM ( EinkPreis )
FROM lager;
```

```
SELECT SUM ( EinkPreis )
FROM lager
WHERE EinkPreis > 100
AND EinkPreis < 200;
```

Um eine Mittelwertberechnung aus einer Menge von Werten durchzuführen wird AVG benutzt:

```
SELECT  AVG ( EinkPreis )
FROM lager;
```

```
SELECT AVG ( EinkPreis )
FROM lager
WHERE EinkPreis > 100
AND EinkPreis < 200;
```

Um den größten oder kleinsten Wert von Spalten zu bestimmen, wird MIN oder MAX verwendet:

```
SELECT  MIN ( EinkPreis )
FROM lager;
```

```
SELECT MAX ( EinkPreis )
FROM lager;
```

Arbeiten mit Tabellenverbunden

In den voran gegangenen Abschnitten wurde immer nur mit einer Tabelle gearbeitet. Im weiteren Verlauf sollen nun mehrere Tabellen dazu verwendet werden, um Abfragen zu generieren.

Um Daten aus verschiedenen Tabellen zusammen abzufragen, müssen zwischen diesen Tabellen Beziehungen eingerichtet worden sein, wobei die JOIN-Schlüssel meistens die primären Schlüssel sind.

Anhand der gemeinsamen Werte in den entsprechenden Spalten (meistens Primär- oder Fremdschlüssel), können Zeilen einer Tabelle mit Zeilen einer anderen Tabelle verknüpft werden.

Datenbanken bestehen meistens aus vielen Tabellen, aus denen Daten abgefragt