

ShopDB – SQL-Übungspaket

NordTech GmbH Onlineshop • Tabellen anlegen, Abfragen prüfen, Abfragen selbst schreiben

Die NordTech GmbH aus Kiel betreibt einen kleinen Onlineshop für IT-Hardware und Software. Sie bauen die Datenbank dazu selbst auf und arbeiten anschließend mit ihr.

Ablauf

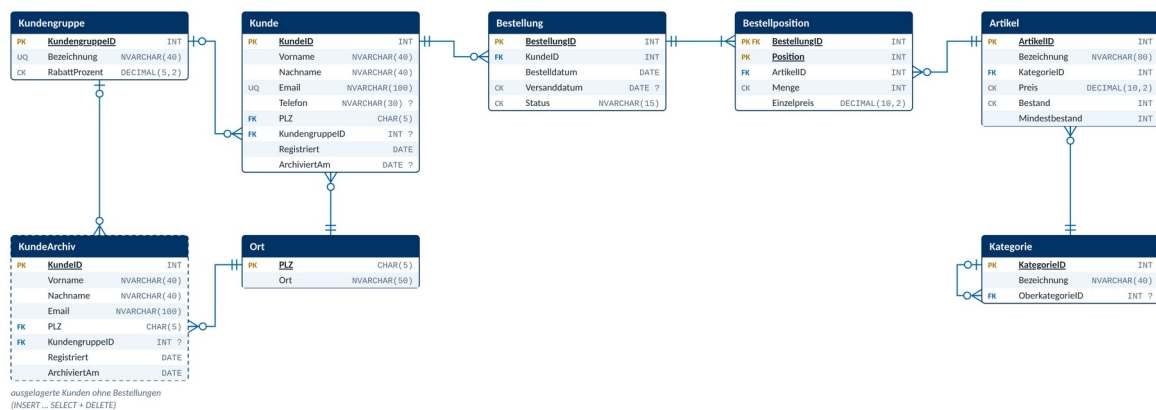
Stufe	Was Sie tun	Kontrolle
1	Tabellen nach Spezifikation anlegen, danach das Datenskript ausführen	Skript läuft ohne Fehler
2	Vorgegebene Abfragen ausführen und verstehen	Ergebnis = Sollergebnis
3	Abfragen zu neuen Fragestellungen selbst schreiben	Ergebnis = Sollergebnis

Dateien

- 02_daten.sql – füllt Ihre Tabellen mit den Übungsdaten (Stufe 1)
- 03_uebungsabfragen.sql – alle Abfragen aus Stufe 2

Datenmodell

Das ER-Diagramm zeigt alle acht Tabellen und ihre Beziehungen in Crow's-Foot-Notation. Ein Fragezeichen hinter dem Datentyp bedeutet: NULL erlaubt.



Stufe 1: Tabellen anlegen

Sie schreiben die CREATE-Anweisungen selbst. Damit das vorgegebene Datenskript danach funktioniert, gelten feste Vorgaben:

- Tabellen- und Spaltennamen genau wie angegeben.
- Datentypen genau wie angegeben.
- **Pflicht = ja** bedeutet NOT NULL, **Pflicht = nein** bedeutet NULL erlaubt.
- „**automatisch fortlaufend ab 1**“ heißt: Der Wert wird von der Datenbank vergeben. Das Datenskript enthält dafür keine Werte.
- Jede Regel in der Spalte „Regel“ muss als Einschränkung umgesetzt werden. In Stufe 2, Teil B wird das geprüft.

Ort – Postleitzahlen und Orte

Spalte	Datentyp	Pflicht	Regel
PLZ	CHAR (5)	ja	Primärschlüssel
Ort	NVARCHAR (50)	ja	

Kundengruppe – Gruppen mit Rabatt

Spalte	Datentyp	Pflicht	Regel
KundengruppeID	INT	ja	Primärschlüssel, automatisch fortlaufend ab 1
Bezeichnung	NVARCHAR (40)	ja	eindeutig
RabattProzent	DECIMAL (5,2)	ja	Standardwert 0; nur Werte von 0 bis 50

Kunde – aktive und per Datum archivierte Kunden

Spalte	Datentyp	Pflicht	Regel
KundeID	INT	ja	Primärschlüssel, automatisch fortlaufend ab 1
Vorname	NVARCHAR (40)	ja	
Nachname	NVARCHAR (40)	ja	
Email	NVARCHAR (100)	ja	eindeutig
Telefon	NVARCHAR (30)	nein	
PLZ	CHAR (5)	ja	Fremdschlüssel auf Ort
KundengruppeID	INT	nein	Fremdschlüssel auf Kundengruppe
Registriert	DATE	ja	Standardwert: heutiges Datum
ArchiviertAm	DATE	nein	leer = aktiver Kunde

KundeArchiv – ausgelagerte Kunden ohne Bestellungen

Spalte	Datentyp	Pflicht	Regel
KundeID	INT	ja	Primärschlüssel, NICHT automatisch (wird aus Kunde übernommen)
Vorname	NVARCHAR(40)	ja	
Nachname	NVARCHAR(40)	ja	
Email	NVARCHAR(100)	ja	
PLZ	CHAR(5)	ja	Fremdschlüssel auf Ort
KundengruppeID	INT	nein	Fremdschlüssel auf Kundengruppe
Registriert	DATE	ja	
ArchiviertAm	DATE	ja	

Kategorie – Artikelkategorien mit Oberkategorie

Spalte	Datentyp	Pflicht	Regel
KategorieID	INT	ja	Primärschlüssel, automatisch fortlaufend ab 1
Bezeichnung	NVARCHAR(40)	ja	
OberkategorieID	INT	nein	Fremdschlüssel auf Kategorie (dieselbe Tabelle)

Artikel – Sortiment

Spalte	Datentyp	Pflicht	Regel
ArtikelID	INT	ja	Primärschlüssel, automatisch fortlaufend ab 1
Bezeichnung	NVARCHAR(80)	ja	
KategorieID	INT	ja	Fremdschlüssel auf Kategorie
Preis	DECIMAL(10,2)	ja	muss größer als 0 sein
Bestand	INT	ja	Standardwert 0; darf nicht negativ sein
Mindestbestand	INT	ja	Standardwert 0

Bestellung – Bestellkopf

Spalte	Datentyp	Pflicht	Regel
BestellungID	INT	ja	Primärschlüssel, automatisch fortlaufend ab 1
KundeID	INT	ja	Fremdschlüssel auf Kunde
Bestelldatum	DATE	ja	Standardwert: heutiges Datum
Versanddatum	DATE	nein	darf nicht vor dem Bestelldatum liegen
Status	NVARCHAR(15)	ja	Standardwert 'offen'; nur 'offen', 'versendet' oder 'storniert'

Bestellposition – Positionen einer Bestellung

Spalte	Datentyp	Pflicht	Regel
BestellungID	INT	ja	Fremdschlüssel auf Bestellung; wird eine Bestellung gelöscht, werden ihre Positionen mitgelöscht
Position	INT	ja	Primärschlüssel zusammen mit BestellungID
ArtikelID	INT	ja	Fremdschlüssel auf Artikel
Menge	INT	ja	muss größer als 0 sein
Einzelpreis	DECIMAL (10,2)	ja	Preis zum Bestellzeitpunkt

Aufgaben

Aufgabe 1.1: Reihenfolge festlegen

Legen Sie fest, in welcher Reihenfolge die acht Tabellen angelegt werden müssen. Begründen Sie die Reihenfolge in zwei bis drei Sätzen.

Aufgabe 1.2: Datenbank und Tabellen anlegen

Legen Sie die Datenbank ShopDB an und erstellen Sie alle acht Tabellen genau nach der Spezifikation. Tabellen- und Spaltennamen, Datentypen und Regeln müssen exakt übernommen werden, sonst läuft das Datenskript nicht.

Aufgabe 1.3: Index anlegen

Legen Sie auf der Spalte Nachname der Tabelle Kunde einen Index mit dem Namen IX_Kunde_Nachname an.

Aufgabe 1.4: Daten einspielen

Führen Sie das Skript 02_daten.sql aus. Es muss ohne Fehlermeldung durchlaufen. Bei einem Fehler: Meldung lesen, CREATE-Anweisung korrigieren, Datenbank löschen, neu anlegen und das Skript erneut ausführen.

Achtung: Das Datenskript darf nur einmal in eine leere Datenbank laufen. Läuft es zweimal oder nach einem Fehler erneut, stimmen die automatisch vergebenen IDs nicht mehr. Dann die Datenbank löschen und von vorn beginnen.

Stufe 2: Vorgegebene Abfragen

Alle Abfragen stehen in 03_uebungsabfragen.sql. Markieren Sie jeweils eine Abfrage, führen Sie sie aus und vergleichen Sie Ihr Ergebnis mit dem Sollergebnis. Überlegen Sie bei jeder Abfrage, warum genau diese Zeilen herauskommen.

Teil A: Datenbank kontrollieren

Diese Abfrage zählt die Datensätze jeder Tabelle. Stimmt eine Zahl nicht, ist beim Anlegen oder beim Datenskript etwas schiefgegangen.

```
SELECT 'Ort' AS Tabelle, COUNT(*) AS Anzahl FROM Ort
UNION ALL SELECT 'Kundengruppe', COUNT(*) FROM Kundengruppe
UNION ALL SELECT 'Kunde', COUNT(*) FROM Kunde
UNION ALL SELECT 'KundeArchiv', COUNT(*) FROM KundeArchiv
UNION ALL SELECT 'Kategorie', COUNT(*) FROM Kategorie
UNION ALL SELECT 'Artikel', COUNT(*) FROM Artikel
UNION ALL SELECT 'Bestellung', COUNT(*) FROM Bestellung
UNION ALL SELECT 'Bestellposition', COUNT(*) FROM Bestellposition;
```

Sollergebnis · 8 Zeilen

Tabelle	Anzahl
Ort	8
Kundengruppe	4
Kunde	10
KundeArchiv	2
Kategorie	6
Artikel	11
Bestellung	14
Bestellposition	26

Teil B: Regeln testen

Jede dieser Anweisungen muss mit einer Fehlermeldung abbrechen. Läuft eine ohne Fehler durch, fehlt die passende Regel in Ihrer CREATE-Anweisung. Korrigieren Sie dann Ihre Tabelle. Jede Anweisung steht in der Datei zwischen BEGIN TRANSACTION und ROLLBACK, die Daten bleiben also in jedem Fall unverändert.

Test	Was passiert	Geprüfte Regel	Soll
B1	Negativer Preis	Preis > 0	Fehler 547 (CHECK)
B2	Doppelte E-Mail	Email eindeutig	Fehler 2627 (UNIQUE)
B3	Unbekannte PLZ	PLZ muss in Ort existieren	Fehler 547 (FOREIGN KEY)
B4	Ungültiger Status	nur offen/versendet/storniert	Fehler 547 (CHECK)
B5	Versand vor Bestellung	Versanddatum >= Bestelldatum	Fehler 547 (CHECK)
B6	Rabatt zu hoch	Rabatt 0 bis 50	Fehler 547 (CHECK)
B7	Position doppelt	BestellungID + Position eindeutig	Fehler 2627 (PRIMARY KEY)
B8	Kunde mit Bestellungen löschen	Bestellung verweist auf Kunde	Fehler 547 (FOREIGN KEY)

Teil C: Übungsabfragen

Dezimalzahlen sind hier mit zwei Nachkommastellen angegeben. SQL Server zeigt bei Berechnungen manchmal mehr Stellen an, zum Beispiel 277.960000 statt 277.96. Das gilt als gleiches Ergebnis.

C1: Spalten wählen und sortieren

SELECT, Alias, ORDER BY · Alle Kunden alphabetisch nach Nachname, die E-Mail-Spalte erhält eine neue Überschrift.

```
SELECT Nachname, Vorname, Email AS [E-Mail]
FROM Kunde
ORDER BY Nachname, Vorname;
```

Sollergebnis · 10 Zeilen

Nachname	Vorname	E-Mail
Fischer	Klara	k.fischer@web.de
Jensen	Ole	ole.jensen@mail.de
Krüger	Jana	einkauf@balticport.de
Meier	Sabine	s.meier@nordtech.de
Müller	Max	max.mueller@web.de
Petersen	Lars	l.petersen@gmx.de
Richter	Tom	tom.richter@mail.de
Schmidt	Anna	anna.schmidt@mail.de
Wagner	Mia	verwaltung@hanse-gym.de
Wolf	Paul	p.wolf@gmx.de

C2: Textmuster

WHERE, LIKE · Kunden mit einer E-Mail-Adresse bei web.de.

```
SELECT Nachname, Email
FROM Kunde
WHERE Email LIKE '%@web.de'
ORDER BY Nachname;
```

Sollergebnis · 2 Zeilen

Nachname	Email
Fischer	k.fischer@web.de
Müller	max.mueller@web.de

C3: Wertebereich und Liste

BETWEEN, IN · Artikel zwischen 100 und 300 Euro; danach alle Kunden aus zwei bestimmten Postleitzahlen.

```
SELECT Bezeichnung, Preis
FROM Artikel
WHERE Preis BETWEEN 100 AND 300
ORDER BY Preis;
```

Sollergebnis · 3 Zeilen

Bezeichnung	Preis
Monitor 24 Zoll FHD	159.00
USB-C Dockingstation	179.00
Monitor 27 Zoll WQHD	289.00

```
SELECT Nachname, PLZ
FROM Kunde
WHERE PLZ IN ('23562', '24103')
ORDER BY Nachname;
```

Sollergebnis · 2 Zeilen

Nachname	PLZ
Meier	24103
Wagner	23562

C4: Fehlende Werte

IS NULL, ISNULL · Kunden ohne Telefonnummer. Achtung: WHERE Telefon = NULL liefert nie ein Ergebnis.

```
SELECT Nachname, ISNULL(Telefon, 'keine Angabe') AS Telefon
FROM Kunde
WHERE Telefon IS NULL
ORDER BY Nachname;
```

Sollergebnis · 4 Zeilen

Nachname	Telefon
Jensen	keine Angabe
Müller	keine Angabe
Richter	keine Angabe
Wolf	keine Angabe

C5: Fallunterscheidung

CASE · Lagerstatus je Artikel aus Bestand und Mindestbestand ableiten.

```
SELECT Bezeichnung, Bestand, Mindestbestand,
       CASE WHEN Bestand = 0          THEN 'ausverkauft'
            WHEN Bestand < Mindestbestand THEN 'nachbestellen'
            ELSE 'ok' END             AS Lagerstatus
FROM   Artikel
ORDER BY ArtikelID;
```

Sollergebnis · 11 Zeilen

Bezeichnung	Bestand	Mindestbestand	Lagerstatus
Notebook ProBook 14	12	5	ok
Notebook UltraLite 13	3	5	nachbestellen
Monitor 24 Zoll FHD	25	10	ok
Monitor 27 Zoll WQHD	8	5	ok
Maus kabellos	120	30	ok
Tastatur DE-Layout	4	20	nachbestellen
USB-C Dockingstation	15	5	ok
Office-Paket Jahreslizenz	999	0	ok
Antivirus 3 Geräte	999	0	ok
Headset Basic	0	10	ausverkauft
Webcam HD	30	10	ok

C6: Zählen und Rechnen

COUNT, MIN, MAX, AVG · COUNT(*) zählt alle Zeilen, COUNT(Spalte) nur Zeilen ohne NULL.

```
SELECT COUNT(*)          AS AnzahlKunden,
       COUNT(Telefon)     AS MitTelefon,
       COUNT(KundengruppeID) AS MitGruppe
FROM   Kunde;
```

Sollergebnis · 1 Zeile

AnzahlKunden	MitTelefon	MitGruppe
10	6	9

```
SELECT MIN(Preis) AS Minimum, MAX(Preis) AS Maximum,
       ROUND(AVG(Preis), 2) AS Durchschnitt
FROM   Artikel;
```

Sollergebnis · 1 Zeile

Minimum	Maximum	Durchschnitt
19.90	1249.00	277.96

C7: Gruppieren und filtern

GROUP BY, HAVING · Kunden-IDs mit mehr als einer Bestellung. HAVING filtert nach dem Gruppieren, WHERE davor.

```
SELECT KundeID, COUNT(*) AS AnzahlBestellungen
FROM   Bestellung
GROUP BY KundeID
HAVING COUNT(*) > 1
ORDER BY KundeID;
```

Sollergebnis · 5 Zeilen

KundeID	AnzahlBestellungen
1	2
2	3
3	2
5	2
9	2

C8: Zwei Tabellen verbinden

INNER JOIN · Kunden mit dem Ortsnamen aus der Tabelle Ort.

```
SELECT k.Nachname, o.Ort
FROM   Kunde k
INNER JOIN Ort o ON o.PLZ = k.PLZ
ORDER BY o.Ort, k.Nachname;
```

Sollergebnis · 10 Zeilen

Nachname	Ort
Richter	Bad Oldesloe
Petersen	Flensburg
Fischer	Hamburg
Meier	Kiel
Krüger	Lübeck
Müller	Lübeck
Schmidt	Lübeck
Wagner	Lübeck
Jensen	Neustadt in Holstein
Wolf	Rostock

C9: Vier Tabellen verbinden

JOIN über mehrere Tabellen · Was steckt in Bestellung 4? Der Weg führt von der Bestellung über die Positionen zum Artikel.

```
SELECT b.BestellungID, k.Nachname, bp.Position,
       a.Bezeichnung, bp.Menge, bp.Einzelpreis
FROM   Bestellung b
JOIN   Kunde k          ON k.KundeID = b.KundeID
JOIN   Bestellposition bp ON bp.BestellungID = b.BestellungID
JOIN   Artikel a        ON a.ArtikelID = bp.ArtikelID
WHERE  b.BestellungID = 4
ORDER BY bp.Position;
```

Sollergebnis · 3 Zeilen

BestellungID	Nachname	Position	Bezeichnung	Menge	Einzelpreis
4	Meier	1	Notebook ProBook 14	10	899.00
4	Meier	2	Monitor 24 Zoll FHD	10	159.00
4	Meier	3	Office-Paket Jahreslizenz	10	99.00

C10: Datensätze ohne Partner

LEFT JOIN, RIGHT JOIN · Kunden, die noch nie bestellt haben. Beide Abfragen liefern dasselbe, nur die Tabellenreihenfolge ist vertauscht.

```
SELECT k.KundeID, k.Nachname
FROM   Kunde k
LEFT JOIN Bestellung b ON b.KundeID = k.KundeID
WHERE  b.BestellungID IS NULL
ORDER BY k.KundeID;
```

Sollergebnis · 2 Zeilen

KundeID	Nachname
8	Wolf
10	Richter

```
SELECT k.KundeID, k.Nachname
FROM   Bestellung b
RIGHT JOIN Kunde k ON k.KundeID = b.KundeID
WHERE  b.BestellungID IS NULL
ORDER BY k.KundeID;
```

Sollergebnis · 2 Zeilen

KundeID	Nachname
8	Wolf
10	Richter

C11: Tabelle mit sich selbst verbinden

Self-Join · Jede Kategorie mit ihrer Oberkategorie. Dieselbe Tabelle bekommt zwei Aliasnamen.

```
SELECT u.Bezeichnung AS Kategorie, o.Bezeichnung AS Oberkategorie
FROM   Kategorie u
LEFT JOIN Kategorie o ON o.KategorieID = u.OberkategorieID
ORDER BY u.KategorieID;
```

Sollergebnis · 6 Zeilen

Kategorie	Oberkategorie
Hardware	NULL
Software	NULL
Notebooks	Hardware
Monitore	Hardware
Zubehör	Hardware
Lizenzen	Software

C12: Unterabfrage mit Einzelwert

Unterabfrage · Artikel, die teurer sind als der Durchschnittspreis aller Artikel.

```
SELECT Bezeichnung, Preis
FROM   Artikel
WHERE  Preis > (SELECT AVG(Preis) FROM Artikel)
ORDER BY Preis DESC;
```

Sollergebnis · 3 Zeilen

Bezeichnung	Preis
Notebook UltraLite 13	1249.00
Notebook ProBook 14	899.00
Monitor 27 Zoll WQHD	289.00

C13: Gibt es keinen passenden Datensatz?

NOT EXISTS · Artikel, die in keiner Bestellposition vorkommen.

```
SELECT a.ArtikelID, a.Bezeichnung
FROM   Artikel a
WHERE  NOT EXISTS (SELECT 1 FROM Bestellposition bp
                    WHERE bp.ArtikelID = a.ArtikelID)
ORDER BY a.ArtikelID;
```

Sollergebnis · 2 Zeilen

ArtikelID	Bezeichnung
10	Headset Basic
11	Webcam HD

C14: Datumsteile

YEAR, MONTH, GROUP BY · Anzahl Bestellungen je Jahr und Monat.

```
SELECT YEAR(Bestelldatum) AS Jahr,  
       MONTH(Bestelldatum) AS Monat,  
       COUNT(*) AS Anzahl  
FROM   Bestellung  
GROUP BY YEAR(Bestelldatum), MONTH(Bestelldatum)  
ORDER BY Jahr, Monat;
```

Sollergebnis · 8 Zeilen

Jahr	Monat	Anzahl
2025	8	1
2025	11	2
2026	1	1
2026	2	1
2026	3	2
2026	6	2
2026	9	2
2026	10	3

C15: Ergebnisse untereinander hängen

UNION ALL mit JOIN · Aktive und ausgelagerte Kunden in einer Liste, jeweils mit Ortsnamen. Spaltenanzahl und Datentypen müssen in beiden Teilen passen.

```
SELECT k.KundeID, k.Nachname, o.Ort, 'Kunde' AS Quelle
FROM   Kunde k
JOIN   Ort o ON o.PLZ = k.PLZ
UNION ALL
SELECT a.KundeID, a.Nachname, o.Ort, 'Archiv'
FROM   KundeArchiv a
JOIN   Ort o ON o.PLZ = a.PLZ
ORDER BY KundeID;
```

Sollergebnis · 12 Zeilen

KundeID	Nachname	Ort	Quelle
1	Schmidt	Lübeck	Kunde
2	Müller	Lübeck	Kunde
3	Meier	Kiel	Kunde
4	Petersen	Flensburg	Kunde
5	Wagner	Lübeck	Kunde
6	Jensen	Neustadt in Holstein	Kunde
7	Fischer	Hamburg	Kunde
8	Wolf	Rostock	Kunde
9	Krüger	Lübeck	Kunde
10	Richter	Bad Oldesloe	Kunde
11	Sommer	Lübeck	Archiv
12	Koch	Kiel	Archiv

C16: Ändern auf Probe

UPDATE in Transaktion, ROLLBACK · Monitore werden testweise 5 % teurer. ROLLBACK macht die Änderung rückgängig, danach gelten wieder die alten Preise.

```
BEGIN TRANSACTION;
  UPDATE Artikel SET Preis = Preis * 1.05 WHERE KategorieID = 4;
  SELECT Bezeichnung, Preis FROM Artikel
  WHERE KategorieID = 4 ORDER BY ArtikelID;
ROLLBACK;
SELECT Bezeichnung, Preis FROM Artikel
WHERE KategorieID = 4 ORDER BY ArtikelID;
```

Sollergebnis: Erste Ausgabe (in der Transaktion) · 2 Zeilen

Bezeichnung	Preis
Monitor 24 Zoll FHD	166.95
Monitor 27 Zoll WQHD	303.45

Sollergebnis: Zweite Ausgabe (nach ROLLBACK) · 2 Zeilen

Bezeichnung	Preis
Monitor 24 Zoll FHD	159.00
Monitor 27 Zoll WQHD	289.00

Stufe 3: Abfragen selbst schreiben

Jetzt schreiben Sie die Abfragen selbst. Die Fragestellungen ähneln Stufe 2, sind aber nicht gleich. Prüfen Sie Ihr Ergebnis mit dem Sollergebnis: gleiche Zeilen, gleiche Werte, gleiche Reihenfolge. Spaltennamen dürfen abweichen, die Spaltenreihenfolge nicht.

- Bearbeiten Sie die Aufgaben der Reihe nach.
- Die Aufgaben 3.22 und 3.23 verändern Daten. Bearbeiten Sie sie zuletzt.
- Wo keine Sortierung verlangt ist, kann die Zeilenreihenfolge abweichen.

Aufgabe 3.1: ORDER BY

Geben Sie alle Artikel mit Bezeichnung und Preis aus, der teuerste Artikel zuerst.

Sollergebnis · 11 Zeilen

Bezeichnung	Preis
Notebook UltraLite 13	1249.00
Notebook ProBook 14	899.00
Monitor 27 Zoll WQHD	289.00
USB-C Dockingstation	179.00
Monitor 24 Zoll FHD	159.00
Office-Paket Jahreslizenz	99.00
Webcam HD	59.00
Antivirus 3 Geräte	39.90
Tastatur DE-Layout	34.90
Headset Basic	29.90
Maus kabellos	19.90

Aufgabe 3.2: LIKE

Welche Artikel haben eine Bezeichnung, die mit „Notebook“ beginnt? Ausgabe: ArtikelID und Bezeichnung, sortiert nach ArtikelID.

Sollergebnis · 2 Zeilen

ArtikelID	Bezeichnung
1	Notebook ProBook 14
2	Notebook UltraLite 13

Aufgabe 3.3: BETWEEN mit Datum

Listen Sie alle Bestellungen aus dem ersten Quartal 2026 (1. Januar bis 31. März) mit BestellungsID, KundeID und Bestelldatum auf, sortiert nach Bestelldatum.

Sollergebnis · 4 Zeilen

BestellungsID	KundeID	Bestelldatum
4	3	2026-01-12
5	2	2026-02-03
6	9	2026-03-02
7	5	2026-03-15

Aufgabe 3.4: IS NULL

Welche Bestellungen wurden noch nicht versendet? Ausgabe: BestellungsID, Bestelldatum und Status, sortiert nach BestellungsID.

Sollergebnis · 3 Zeilen

BestellungsID	Bestelldatum	Status
12	2026-10-05	storniert
13	2026-10-20	offen
14	2026-10-23	offen

Aufgabe 3.5: IS NULL beim Fremdschlüssel

Welche Kunden sind keiner Kundengruppe zugeordnet? Ausgabe: Vorname und Nachname.

Sollergebnis · 1 Zeile

Vorname	Nachname
Ole	Jensen

Aufgabe 3.6: CASE

Geben Sie alle Kunden mit Nachname und einer Spalte Kundenstatus aus. Sie soll „archiviert“ enthalten, wenn ein Archivierungsdatum eingetragen ist, sonst „aktiv“. Sortierung nach Nachname.

Sollergebnis · 10 Zeilen

Nachname	Kundenstatus
Fischer	archiviert
Jensen	aktiv
Krüger	aktiv
Meier	aktiv
Müller	aktiv
Petersen	aktiv
Richter	aktiv
Schmidt	aktiv
Wagner	aktiv
Wolf	archiviert

Aufgabe 3.7: Aggregatfunktionen

Ermitteln Sie in einer Abfrage die Anzahl der Artikel, den gesamten Lagerwert (Preis mal Bestand, über alle Artikel summiert) und den höchsten Bestand.

Sollergebnis · 1 Zeile

AnzahlArtikel	Lagerwert	HoechsterBestand
11	166565.70	999

Aufgabe 3.8: JOIN, GROUP BY, HAVING

Wie viele Artikel gibt es je Kategorie? Geben Sie den Kategorienamen und die Anzahl aus, aber nur für Kategorien mit mindestens drei Artikeln.

Sollergebnis · 1 Zeile

Kategorie	AnzahlArtikel
Zubehör	5

Aufgabe 3.9: JOIN über drei Tabellen

Geben Sie für alle offenen Bestellungen BestellungsID, Bestelldatum, den Nachnamen des Kunden und seinen Ort aus, sortiert nach BestellungsID.

Sollergebnis · 2 Zeilen

BestellungsID	Bestelldatum	Nachname	Ort
13	2026-10-20	Meier	Kiel
14	2026-10-23	Jensen	Neustadt in Holstein

Aufgabe 3.10: JOIN, Berechnung, GROUP BY

Berechnen Sie den Umsatz je Kunde (Menge mal Einzelpreis) ohne stornierte Bestellungen. Ausgabe: Nachname und Umsatz, höchster Umsatz zuerst.

Sollergebnis · 7 Zeilen

Nachname	Umsatz
Meier	14204.60
Wagner	7748.50
Krüger	5986.70
Müller	1511.10
Fischer	1249.00
Schmidt	779.00
Jensen	174.95

Aufgabe 3.11: LEFT JOIN mit COUNT

Wie viele Bestellungen hat jeder Kunde? Auch Kunden ohne Bestellung sollen mit der Anzahl 0 erscheinen. Ausgabe: KundeID, Nachname, Anzahl, sortiert nach KundeID.

Tipp: COUNT(*) würde bei Kunden ohne Bestellung 1 liefern.

Sollergebnis · 10 Zeilen

KundeID	Nachname	Anzahl
1	Schmidt	2
2	Müller	3
3	Meier	2
4	Petersen	1
5	Wagner	2
6	Jensen	1
7	Fischer	1
8	Wolf	0
9	Krüger	2
10	Richter	0

Aufgabe 3.12: Self-Join mit weiterem JOIN

Welche Artikel gehören zur Oberkategorie „Software“? Ausgabe: Bezeichnung des Artikels und seine Kategorie, sortiert nach Bezeichnung.

Sollergebnis · 2 Zeilen

Bezeichnung	Kategorie
Antivirus 3 Geräte	Lizenzen
Office-Paket Jahreslizenz	Lizenzen

Aufgabe 3.13: Unterabfrage mit IN

Welche Kunden haben schon einmal einen Artikel der Kategorie „Notebooks“ bestellt? Stornierte Bestellungen zählen nicht. Ausgabe: Vorname und Nachname, sortiert nach Nachname. Lösen Sie die Aufgabe mit einer Unterabfrage.

Sollergebnis · 5 Zeilen

Vorname	Nachname
Klara	Fischer
Jana	Krüger
Sabine	Meier
Max	Müller
Anna	Schmidt

Aufgabe 3.14: NOT EXISTS mit Datum

Welche Kunden haben im Jahr 2026 nichts bestellt? Ausgabe: KundeID und Nachname, sortiert nach KundeID.

Sollergebnis · 3 Zeilen

KundeID	Nachname
7	Fischer
8	Wolf
10	Richter

Aufgabe 3.15: DATEDIFF

Bei welchen versendeten Bestellungen lagen mehr als ein Tag zwischen Bestellung und Versand?
Ausgabe: BestellungID und Lieferzeit in Tagen, sortiert nach BestellungID.

Sollergebnis · 3 Zeilen

BestellungID	Lieferzeit
4	2
7	2
10	2

Aufgabe 3.16: MONTH, JOIN, GROUP BY

Ermitteln Sie den Umsatz je Monat im Jahr 2026 ohne stornierte Bestellungen. Ausgabe: Monat (als Zahl) und Umsatz, sortiert nach Monat.

Sollergebnis · 6 Zeilen

Monat	Umsatz
1	11570.00
2	323.90
3	4872.70
6	3512.40
9	5679.00
10	2809.55

Aufgabe 3.17: YEAR in WHERE

Welche Kunden (Tabelle Kunde) haben sich vor dem Jahr 2021 registriert? Ausgabe: Nachname und Registrierungsdatum, ältestes Datum zuerst.

Sollergebnis · 3 Zeilen

Nachname	Registriert
Fischer	2019-04-10
Meier	2020-01-15
Wolf	2020-02-02

Aufgabe 3.18: UNION mit JOIN

Für eine Mailing-Aktion werden die E-Mail-Adressen aller Geschäftskunden gebraucht, aktiv und ausgelagert. Ausgabe: Nachname und Email, sortiert nach Nachname. Verwenden Sie den Gruppennamen „Geschäftskunde“, nicht die ID.

Sollergebnis · 3 Zeilen

Nachname	Email
Koch	info@koch-buero.de
Krüger	einkauf@balticport.de
Meier	s.meier@nordtech.de

Aufgabe 3.19: UNION ALL mit Summenzeile

Zählen Sie die Bestellungen je Status und hängen Sie eine letzte Zeile „Gesamt“ mit der Gesamtzahl an. Ausgabe: Status und Anzahl. Die Gesamtzeile soll ganz unten stehen.

Tipp: Eine zusätzliche Sortierspalte hilft, die Gesamtzeile nach unten zu bringen.

Sollergebnis · 4 Zeilen

Sortierung	Status	Anzahl
1	offen	2
1	storniert	1
1	versendet	11
2	Gesamt	14

Aufgabe 3.20: VIEW

Erstellen Sie eine View v_Bestellwert mit den Spalten BestellungID, Nachname und Warenwert (Summe aus Menge mal Einzelpreis je Bestellung). Fragen Sie danach über die View alle Bestellungen mit einem Warenwert über 3000 ab, höchster Wert zuerst.

Tipp: CREATE VIEW muss allein in einem Batch stehen. Setzen Sie GO davor und danach.

Sollergebnis · 4 Zeilen

BestellungID	Nachname	Warenwert
4	Meier	11570.00
10	Krüger	5390.00
7	Wagner	4276.00
9	Wagner	3472.50

Aufgabe 3.21: Rabatt berechnen

Berechnen Sie für jede Bestellung von Geschäftskunden den Warenwert und den Betrag nach Abzug des Gruppenrabatts (auf zwei Stellen gerundet). Ausgabe: BestellungID, Nachname, Warenwert, RabattProzent, Endbetrag, sortiert nach BestellungID. Stornierte Bestellungen zählen nicht.

Sollergebnis · 4 Zeilen

BestellungID	Nachname	Warenwert	RabattProzent	Endbetrag
4	Meier	11570.00	10.00	10413.00
6	Krüger	596.70	10.00	537.03
10	Krüger	5390.00	10.00	4851.00
13	Meier	2634.60	10.00	2371.14

Aufgabe 3.22: INSERT ... SELECT, DELETE, Transaktion

Kunde 8 (Wolf) hat nie bestellt und ist bereits als archiviert markiert. Verschieben Sie ihn in die Tabelle KundeArchiv: Datensatz kopieren und anschließend in Kunde löschen. Beides soll in einer Transaktion geschehen. Kontrollieren Sie danach mit einer Abfrage die Zahl der Datensätze in Kunde und KundeArchiv.

Achtung: Diese Aufgabe verändert die Daten dauerhaft. Danach stimmen die Sollergebnisse früherer Aufgaben nicht mehr.

Sollergebnis der Kontrollabfrage · 1 Zeile

Kunde	KundeArchiv
9	3

Aufgabe 3.23: UPDATE mit Unterabfrage

Alle Artikel der Kategorie „Zubehör“ werden um 10 % teurer. Verwenden Sie im UPDATE den Kategorienamen, nicht die ID. Kontrollieren Sie danach die neuen Preise der Kategorie, sortiert nach ArtikelID.

Achtung: Diese Aufgabe verändert die Daten dauerhaft. Danach stimmen die Sollergebnisse früherer Aufgaben nicht mehr.

Sollergebnis der Kontrollabfrage · 5 Zeilen

ArtikelID	Bezeichnung	Preis
5	Maus kabellos	21.89
6	Tastatur DE-Layout	38.39
7	USB-C Dockingstation	196.90
10	Headset Basic	32.89
11	Webcam HD	64.90