



**Hardware and Software**  
Engineered to Work Together

# Oracle Database 12c: SQL Workshop I

Übungen

D80190DE11

Production 1.1 | Dezember 2014 | D88604

Learn more from Oracle University at [oracle.com/education/](http://oracle.com/education/)

**Copyright © 2014, Oracle und/oder verbundene Unternehmen. All rights reserved. Alle Rechte vorbehalten.**

Diese Software und zugehörige Dokumentation werden im Rahmen eines Lizenzvertrages zur Verfügung gestellt, der Einschränkungen hinsichtlich Nutzung und Offenlegung enthält und durch Gesetze zum Schutz geistigen Eigentums geschützt ist. Sofern nicht ausdrücklich in Ihrem Lizenzvertrag vereinbart oder gesetzlich geregelt, darf diese Software weder ganz noch teilweise in irgendeiner Form oder durch irgendein Mittel zu irgendeinem Zweck kopiert, reproduziert, übersetzt, gesendet, verändert, lizenziert, übertragen, verteilt, ausgestellt, ausgeführt, veröffentlicht oder angezeigt werden. Reverse Engineering, Disassemblierung oder Dekompilierung der Software ist verboten, es sei denn, dies ist erforderlich, um die gesetzlich vorgesehene Interoperabilität mit anderer Software zu ermöglichen.

Die hier angegebenen Informationen können jederzeit und ohne vorherige Ankündigung geändert werden. Wir übernehmen keine Gewähr für deren Richtigkeit. Sollten Sie Fehler oder Unstimmigkeiten finden, bitten wir Sie, uns diese schriftlich mitzuteilen.

Wird diese Software oder zugehörige Dokumentation an die Regierung der Vereinigten Staaten von Amerika bzw. einen Lizenznehmer im Auftrag der Regierung der Vereinigten Staaten von Amerika geliefert, gilt Folgendes:

**U.S. GOVERNMENT END USERS:**

Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

Oracle und Java sind eingetragene Marken der Oracle Corporation und/oder ihrer verbundenen Unternehmen. Andere Namen und Bezeichnungen können Marken ihrer jeweiligen Inhaber sein.

## **Autor**

Dimpi Rani Sarmah

## **Technischer Inhalt und Überarbeitung**

Nancy Greenberg, Swarnapriya Shridhar, Bryan Roberts, Laszlo Czinkoczki, KimSeong Loh, Brent Dayley, Jim Spiller, Christopher Wensley, Manish Pawar, Clair Bennett, Yanti Chang, Joel Goodman, Gerlinde Frenzen and Madhavi Siddireddy

**Dieses Buch wurde erstellt mit: Oracle Tutor**

# Inhaltsverzeichnis

---

<b>Übungen zu Lektion 1 – Einführung .....</b>	<b>1-1</b>
Übungen zu Lektion 1 – Überblick .....	1-2
Übung 1 zu Lektion 1 – Einführung .....	1-3
Übung 1 zu Lektion 1 – Lösung: Einführung .....	1-5
<b>Übungen zu Lektion 2 – Daten mit der SQL SELECT-Anweisung abrufen .....</b>	<b>2-1</b>
Übungen zu Lektion 2 – Überblick .....	2-2
Übung 1 zu Lektion 2 – Daten mit der SQL-Anweisung SELECT abrufen .....	2-3
Übung 1 zu Lektion 2 – Lösung: Daten mit der SQL-Anweisung SELECT abrufen .....	2-8
<b>Übungen zu Lektion 3 – Daten einschränken und sortieren .....</b>	<b>3-1</b>
Übungen zu Lektion 3 – Überblick .....	3-2
Übung 1 zu Lektion 3 – Daten einschränken und sortieren .....	3-3
Übung 1 zu Lektion 3 – Lösung: Daten einschränken und sortieren .....	3-7
<b>Übungen zu Lektion 4 – Ausgabe mit Single-Row-Funktionen anpassen .....</b>	<b>4-1</b>
Übungen zu Lektion 4 – Überblick .....	4-2
Übung 1 zu Lektion 4 – Ausgabe mit Single-Row-Funktionen anpassen .....	4-3
Übung 1 zu Lektion 4 – Lösung: Ausgabe mit Single-Row-Funktionen anpassen .....	4-9
<b>Übungen zu Lektion 5 – Konvertierungsfunktionen und bedingte Ausdrücke .....</b>	<b>5-1</b>
Übungen zu Lektion 5 – Überblick .....	5-2
Übung 1 zu Lektion 5 – Konvertierungsfunktionen und bedingte Ausdrücke .....	5-3
Übung 1 zu Lektion 5 – Lösung: Konvertierungsfunktionen und bedingte Ausdrücke .....	5-9
<b>Übungen zu Lektion 6 – Mit Gruppenfunktionen Berichte aus aggregierten Daten erstellen .....</b>	<b>6-1</b>
Übungen zu Lektion 6 – Überblick .....	6-2
Übung 1 zu Lektion 6 – Mit Gruppenfunktionen Berichte zu aggregierten Daten erstellen .....	6-3
Übung 1 zu Lektion 6 – Lösung: Mit Gruppenfunktionen Berichte zu aggregierten Daten erstellen .....	6-6
<b>Übungen zu Lektion 7 – Daten aus mehreren Tabellen mit Joins anzeigen .....</b>	<b>7-1</b>
Übungen zu Lektion 7 – Überblick .....	7-2
Übung 1 zu Lektion 7 – Daten aus mehreren Tabellen mit Joins anzeigen .....	7-3
Übung 1 zu Lektion 7 – Lösung: Daten aus mehreren Tabellen mit Joins anzeigen .....	7-8
<b>Übungen zu Lektion 8 – Unterabfragen in Abfragen .....</b>	<b>8-1</b>
Übungen zu Lektion 8 – Überblick .....	8-2
Übung 1 zu Lektion 8 – Unterabfragen in Abfragen .....	8-3
Übung 1 zu Lektion 8 – Lösung: Unterabfragen in Abfragen .....	8-6
<b>Übungen zu Lektion 9 – Mengenoperatoren .....</b>	<b>9-1</b>
Übungen zu Lektion 9 – Überblick .....	9-2
Übung 1 zu Lektion 9 – Mengenoperatoren .....	9-3
Übung 1 zu Lektion 9 – Lösung: Mengenoperatoren .....	9-5
<b>Übungen zu Lektion 10 – Daten bearbeiten .....</b>	<b>10-1</b>
Übungen zu Lektion 10 – Überblick .....	10-2
Übung 1 zu Lektion 10 – Tabellen mit DML-Anweisungen verwalten .....	10-3
Übung 1 zu Lektion 10 – Lösung: Tabellen mit DML-Anweisungen verwalten .....	10-7
<b>Übungen zu Lektion 11 – Tabellen mit DDL-Anweisungen erstellen und verwalten .....</b>	<b>11-1</b>
Übungen zu Lektion 11 – Überblick .....	11-2
Übung 1 zu Lektion 11 – Data Definition Language – Einführung .....	11-3
Übung 1 zu Lektion 11 – Lösung: Data Definition Language: Einführung .....	11-7

<b>Zusätzliche Übungen und Lösungen .....</b>	<b>12-1</b>
Übung 1 .....	12-2
Übung 1 – Zusätzliche Übungen .....	12-3
Übung 1 – Lösungen: Zusätzliche Übungen .....	12-11
Fallstudie: Onlinebuchhandlung .....	12-16
Übung 2 .....	12-17
Übung 2 – Lösungen.....	12-22

# Übungen zu Lektion 1 – Einführung

## Kapitel 1

# Übungen zu Lektion 1 – Überblick

---

## Übungsüberblick

In dieser Übung starten Sie SQL Developer, erstellen eine neue Datenbankverbindung und navigieren durch die HR-Tabellen. Außerdem legen Sie einige Voreinstellungen für SQL Developer fest.

Die Übungen können weitere Aufgaben mit der Wendung "Falls Sie noch Zeit haben" oder "Wenn Sie eine weitere Herausforderung suchen" enthalten. Bearbeiten Sie diese Aufgaben nur, wenn Sie alle anderen Übungen in der verfügbaren Zeit abgeschlossen haben und Ihre Kenntnisse weiter testen möchten.

Bearbeiten Sie die Übungen langsam und präzise. Sie können mit dem Speichern und Ausführen von Befehlsdateien experimentieren. Wenn Sie Fragen haben, wenden Sie sich an den Dozenten.

## Hinweise

- Für alle schriftlichen Übungen wird Oracle SQL Developer als Entwicklungsumgebung verwendet. Sie können für diesen Kurs zwar auch SQL\*Plus verwenden, die Verwendung von Oracle SQL Developer wird jedoch empfohlen.
- Die Reihenfolge der bei Abfragen aus der Datenbank abgerufenen Zeilen kann sich von der auf den abgebildeten Screenshots unterscheiden.

# Übung 1 zu Lektion 1 – Einführung

---

## Überblick

Dies ist die erste von zahlreichen Übungen in diesem Kurs. Die Lösungen finden Sie (sofern erforderlich) am Ende dieser Übung. Die Übungen decken die meisten Themen ab, die in den entsprechenden Lektionen behandelt wurden.

In dieser Übung führen Sie folgende Aufgaben aus:

- Oracle SQL Developer starten und eine neue Datenbankverbindung für den Account `ora1` erstellen
- Mit Oracle SQL Developer Datenobjekte im Account `ora1` untersuchen. Der Account `ora1` enthält die Tabellen des Schemas `HR`.

Die Übungsdateien befinden sich im Verzeichnis

`/home/oracle/labs/sql1/labs`

Wenn Sie zum Speichern von Übungsdateien aufgefordert werden, speichern Sie die Dateien unter diesem Pfad.

## Aufgaben

1. Starten Sie Oracle SQL Developer über das entsprechende Desktopsymbol.
2. Erstellen Sie in Oracle SQL Developer eine neue Datenbankverbindung.
  - a. Um eine neue Datenbankverbindung zu erstellen, klicken Sie im Connections Navigator mit der rechten Maustaste auf **Connections** und wählen im Kontextmenü die Option **New Connection**. Das Dialogfeld **New/Select Database Connection** wird angezeigt.

- b. Erstellen Sie mit folgenden Informationen eine Datenbankverbindung:

Connection Name: `myconnection`

Username: `ora1`

Password: `ora1`

Hostname: `localhost`

Port: `1521`

SID: `ORCL`

Vergewissern Sie sich, dass das Kontrollkästchen **Save Password** aktiviert ist.

3. Testen Sie die in Oracle SQL Developer erstellte Datenbankverbindung, und melden Sie sich bei der Datenbank an.
  - a. Testen Sie die neue Verbindung.
  - b. Wenn als Status **Success** angezeigt wird, melden Sie sich über diese neue Verbindung bei der Datenbank an.

4. Navigieren Sie im Connections Navigator durch die Tabellen.

a. Zeigen Sie im Connections Navigator die unter dem Knoten **Tables** verfügbaren Objekte an. Vergewissern Sie sich, dass folgende Tabellen vorhanden sind:

COUNTRIES  
DEPARTMENTS  
EMPLOYEES  
JOB\_GRADES  
JOB\_HISTORY  
JOBS  
LOCATIONS  
REGIONS

b. Navigieren Sie durch die Struktur der Tabelle EMPLOYEES.

c. Zeigen Sie die Daten der Tabelle DEPARTMENTS an.



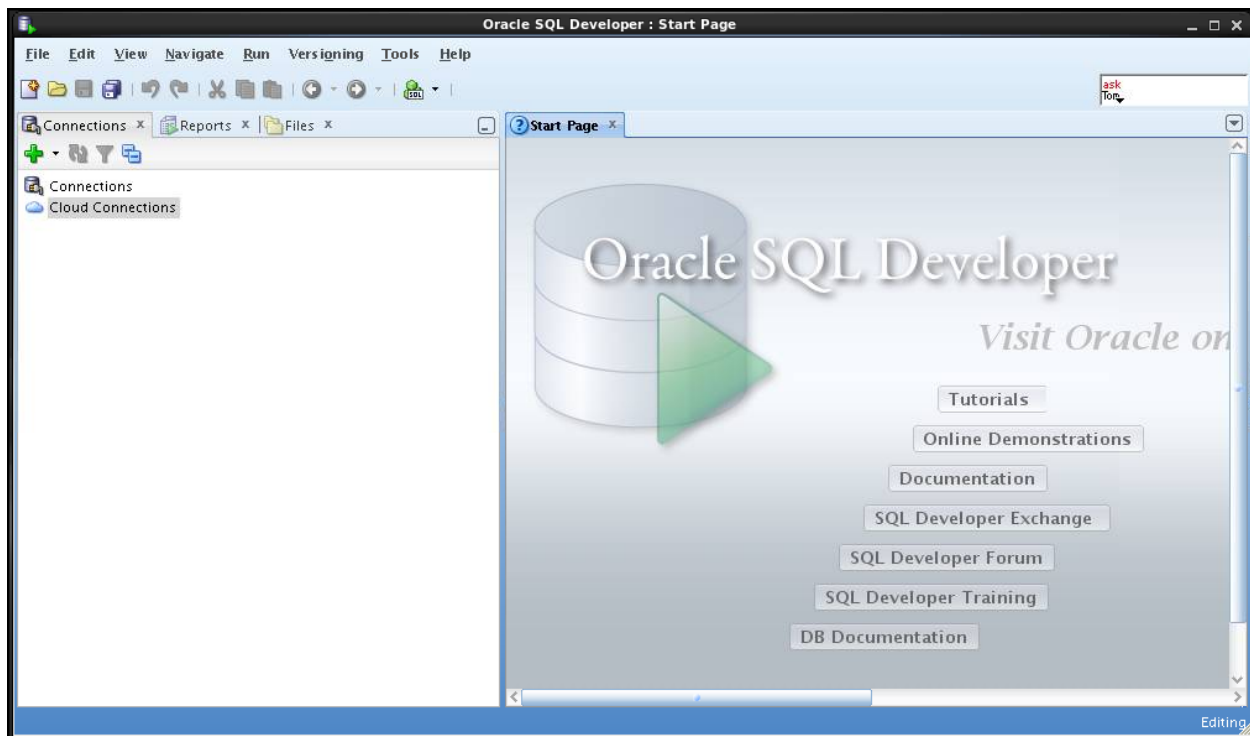
## Übung 1 zu Lektion 1 – Lösung: Einführung

---

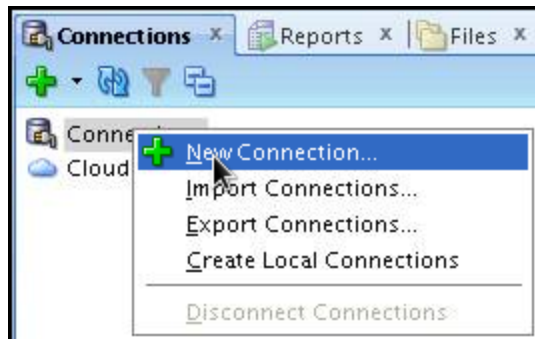
1. Starten Sie Oracle SQL Developer über das entsprechende Desktopsymbol.  
Doppelklicken Sie auf das Desktopsymbol für Oracle SQL Developer.



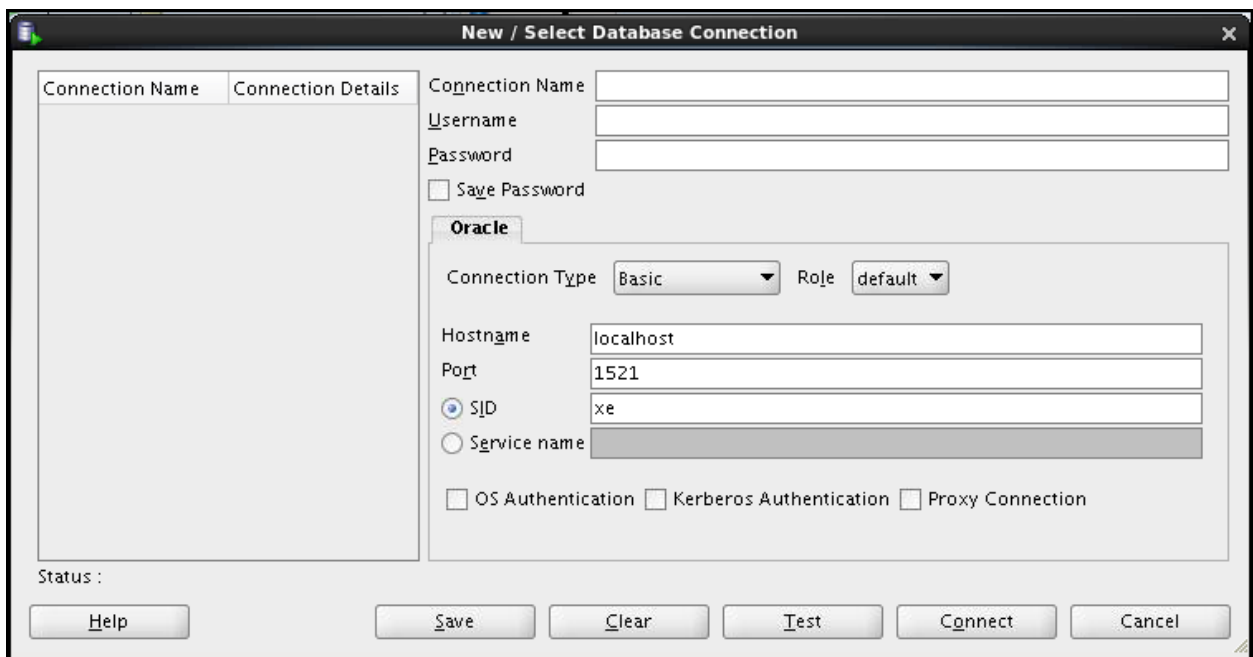
Die Benutzeroberfläche von SQL Developer wird angezeigt.



2. Erstellen Sie in Oracle SQL Developer eine neue Datenbankverbindung.
- a. Um eine neue Datenbankverbindung zu erstellen, klicken Sie im Connections Navigator mit der rechten Maustaste auf **Connections** und wählen im Kontextmenü die Option **New Connection**.



Das Dialogfeld **New/Select Database Connection** wird angezeigt.



- b. Erstellen Sie mit folgenden Informationen eine Datenbankverbindung:
- i. Connection Name: myconnection
  - ii. Username: ora1
  - iii. Password: ora1
  - iv. Hostname: localhost
  - v. Port: 1521
  - vi. SID: ORCL

Vergewissern Sie sich, dass das Kontrollkästchen **Save Password** aktiviert ist.

Connection Name: myconnection  
Username: ora1  
Password: .....  
☒ Save Password

**Oracle**

Connection Type: Basic Role: default  
Hostname: localhost  
Port: 1521  
☒ SID: ORCL  
☐ Service name:   
☐ OS Authentication ☐ Kerberos Authentication ☐ Proxy Connection

Status:   
Help Save Clear Test Connect Cancel

3. Testen Sie die in Oracle SQL Developer erstellte Datenbankverbindung, und melden Sie sich bei der Datenbank an.

a. Testen Sie die neue Verbindung.

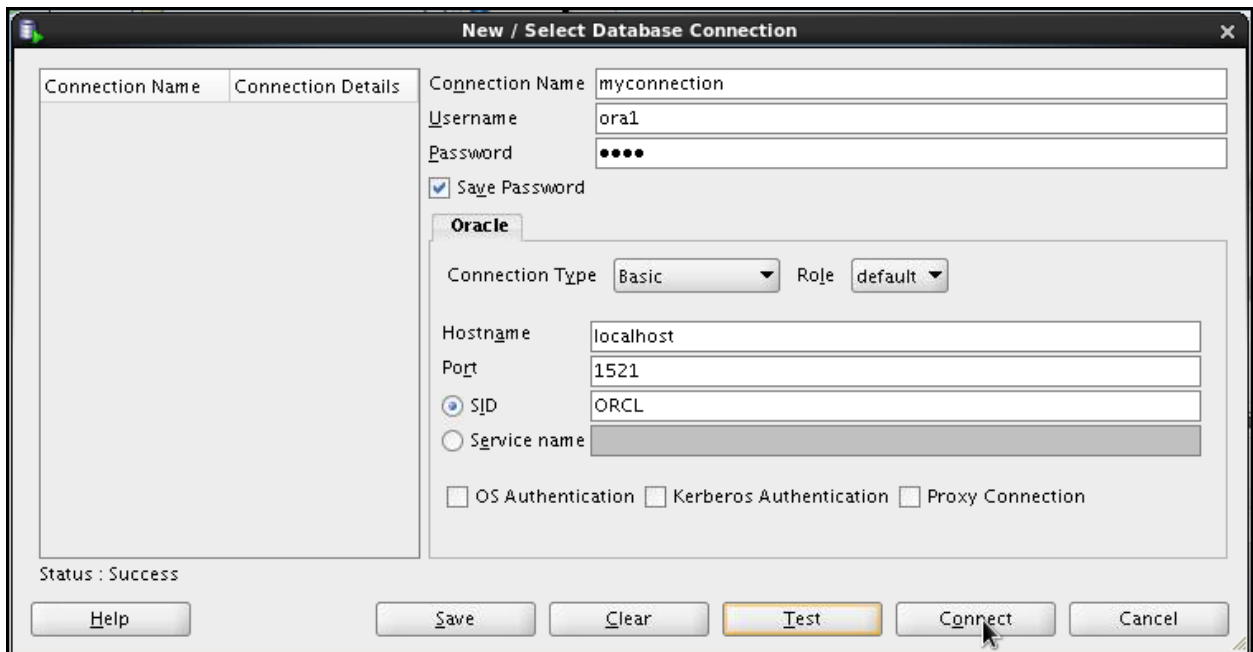
Connection Name: myconnection  
Username: ora1  
Password: .....  
☒ Save Password

**Oracle**

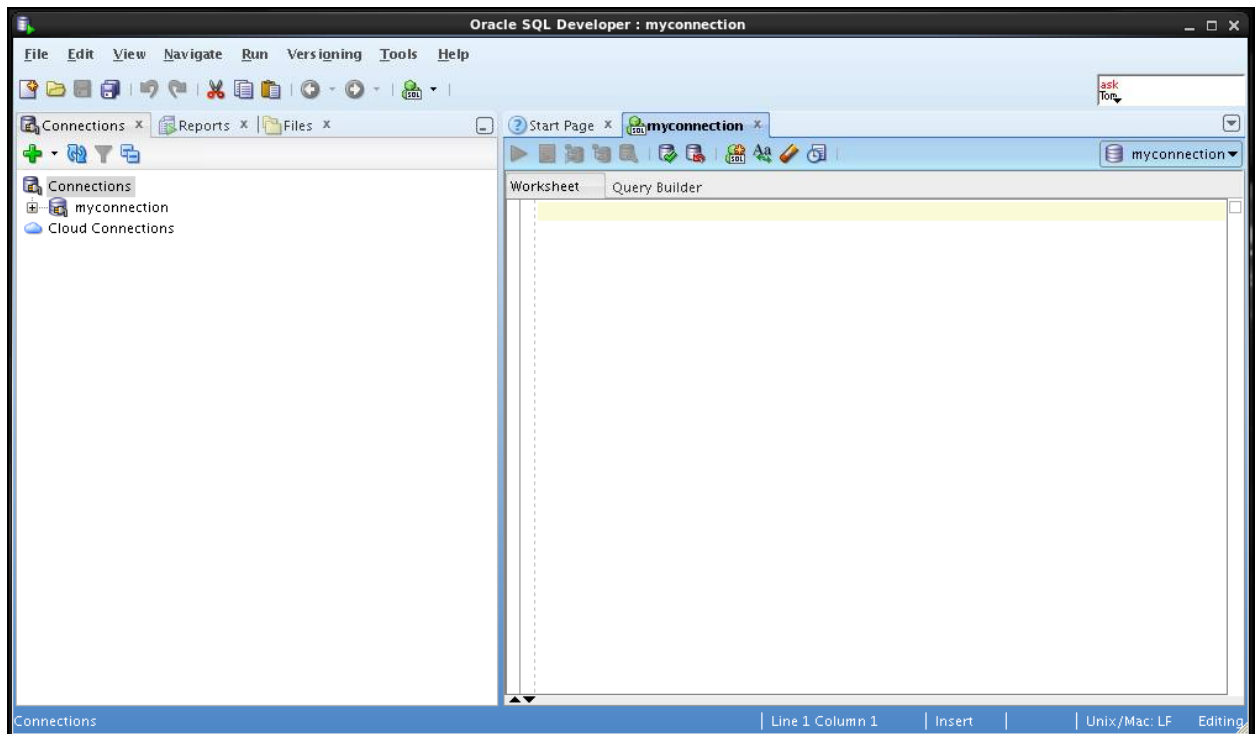
Connection Type: Basic Role: default  
Hostname: localhost  
Port: 1521  
☒ SID: ORCL  
☐ Service name:   
☐ OS Authentication ☐ Kerberos Authentication ☐ Proxy Connection

Status: Success  
Help Save Clear Test Connect Cancel

- b. Wenn als Status **Success** angezeigt wird, melden Sie sich über diese neue Verbindung bei der Datenbank an.



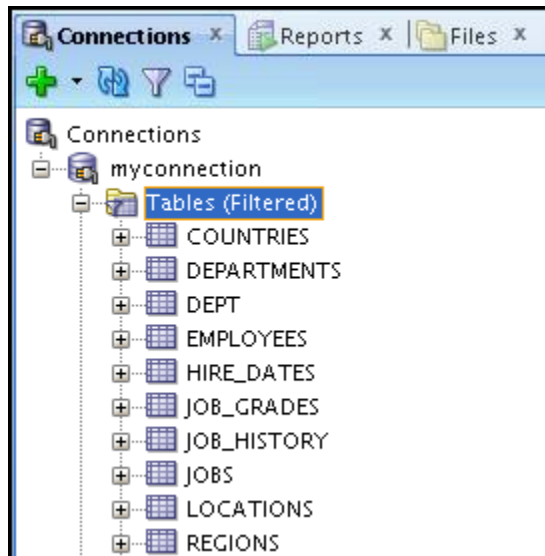
Bei der Erstellung einer Verbindung wird automatisch ein SQL Worksheet für diese Verbindung geöffnet.



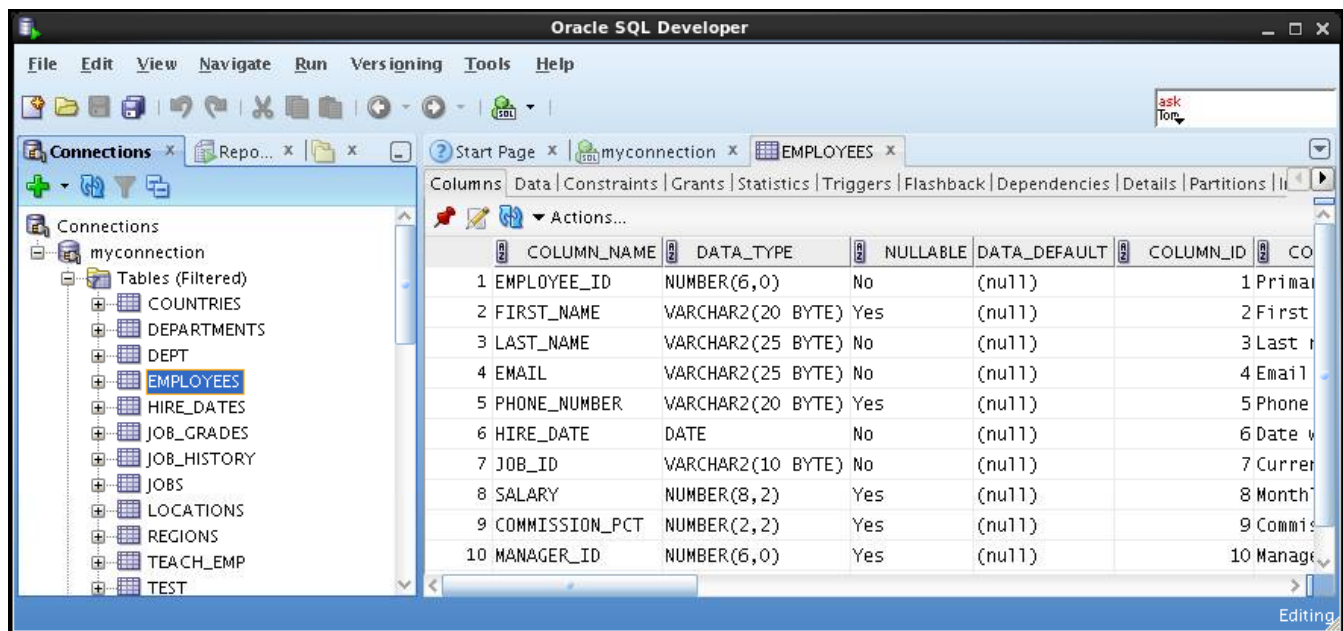
4. Navigieren Sie im Connections Navigator durch die Tabellen.

- a. Zeigen Sie im Connections Navigator die unter dem Knoten **Tables** verfügbaren Objekte an. Vergewissern Sie sich, dass folgende Tabellen vorhanden sind:

COUNTRIES  
DEPARTMENTS  
EMPLOYEES  
JOB\_GRADES  
JOB\_HISTORY  
JOBS  
LOCATIONS  
REGIONS



- b. Navigieren Sie durch die Struktur der Tabelle EMPLOYEES.



c. Zeigen Sie die Daten der Tabelle DEPARTMENTS an.

The screenshot shows the Oracle SQL Developer interface. The title bar reads 'Oracle SQL Developer : Table ORA1.DEPARTMENTS@myconnection'. The main window displays the 'DEPARTMENTS' table data. The left pane shows a tree view of the database schema under 'myconnection', with 'DEPARTMENTS' selected. The right pane shows the 'Data' tab of the table, displaying 8 rows of data. The columns are DEPARTMENT\_ID, DEPARTMENT\_NAME, MANAGER\_ID, and LOCATION\_ID. The data is as follows:

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	10 Administration	200	1700
2	20 Marketing	201	1800
3	50 Shipping	124	1500
4	60 IT	103	1400
5	80 Sales	149	2500
6	90 Executive	100	1700
7	110 Accounting	205	1700
8	190 Contracting	(null)	1700

# **Übungen zu Lektion 2 – Daten mit der SQL- Anweisung `SELECT` abrufen**

## **Kapitel 2**

## Übungen zu Lektion 2 – Überblick

---

### Übungsüberblick

Diese Übungen behandeln folgende Themen:

- Alle Daten aus verschiedenen Tabellen wählen
- Tabellenstrukturen beschreiben
- Arithmetische Berechnungen ausführen und Spaltennamen angeben



# Übung 1 zu Lektion 2 – Daten mit der SQL-Anweisung SELECT abrufen

## Überblick

In dieser Übung erstellen Sie einfache `SELECT`-Abfragen. Dabei verwenden Sie die meisten der in dieser Lektion behandelten `SELECT`-Klauseln und -Vorgänge.

### 1. Aufgabe

Testen Sie Ihre Kenntnisse:

1. Die folgende Anweisung `SELECT` wird erfolgreich ausgeführt:

```
SELECT last_name, job_id, salary AS Sal
FROM   employees;
```

Richtig/Falsch

2. Die folgende Anweisung `SELECT` wird erfolgreich ausgeführt:

```
SELECT *
FROM   job_grades;
```

Richtig/Falsch

3. Die folgende Anweisung enthält vier Codierungsfehler. Können Sie alle vier Fehler finden?

```
SELECT      employee_id, last_name
sal x 12    ANNUAL SALARY
FROM        employees;
```

### 2. Aufgabe

Berücksichtigen Sie folgende Punkte, bevor Sie mit der Übung beginnen:

- Speichern Sie alle Übungsdateien im folgenden Verzeichnis:  
/home/oracle/labs/sql1/labs
- Geben Sie die SQL-Anweisungen in ein SQL Worksheet ein. Achten Sie beim Speichern von Skripten in SQL Developer darauf, dass das erforderliche SQL Worksheet aktiv ist. Wählen Sie im Menü **File** die Option **Save As**, und speichern Sie die SQL-Anweisung als Skript `lab_<lessonno>_<stepno>.sql`. Wenn Sie ein vorhandenes Skript ändern, speichern Sie es mit der Option **Save As** unter einem anderen Dateinamen.
- Um die Abfrage auszuführen, klicken Sie im SQL Worksheet auf das Symbol **Execute Statement**. Alternativ können Sie F9 drücken. Für DML- und DDL-Anweisungen verwenden Sie das Symbol **Run Script** oder drücken F5.
- Nachdem Sie die Abfrage ausgeführt haben, achten Sie darauf, die nächste Abfrage nicht in das gleiche Worksheet einzugeben, sondern ein neues Worksheet zu öffnen.

Sie wurden als SQL-Programmierer bei Acme eingestellt. Ihre erste Aufgabe besteht darin, einige Berichte auf Basis der Daten aus den Human Resources-Tabellen zu erstellen.

4. Prüfen Sie zunächst Struktur und Inhalt der Tabelle `DEPARTMENTS`.

```
DESCRIBE departments
Name          Null    Type
-----
DEPARTMENT_ID NOT NULL  NUMBER(4)
DEPARTMENT_NAME NOT NULL  VARCHAR2(30)
MANAGER_ID    NUMBER(6)
LOCATION_ID     NUMBER(4)
```

	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	10	Administration	200	1700
2	20	Marketing	201	1800
3	50	Shipping	124	1500
4	60	IT	103	1400
5	80	Sales	149	2500
6	90	Executive	100	1700
7	110	Accounting	205	1700
8	190	Contracting	(null)	1700





5. Prüfen Sie Struktur und Inhalt der Tabelle `EMPLOYEES`.

- a. Bestimmen Sie die Struktur der Tabelle `EMPLOYEES`.

```
DESCRIBE employees
Name          Null    Type
-----
EMPLOYEE_ID   NOT NULL  NUMBER(6)
FIRST_NAME    VARCHAR2(20)
LAST_NAME     NOT NULL  VARCHAR2(25)
EMAIL         NOT NULL  VARCHAR2(25)
PHONE_NUMBER  VARCHAR2(20)
HIRE_DATE     NOT NULL  DATE
JOB_ID        NOT NULL  VARCHAR2(10)
SALARY        NUMBER(8,2)
COMMISSION_PCT NUMBER(2,2)
MANAGER_ID    NUMBER(6)
DEPARTMENT_ID NUMBER(4)
```

- b. Die Personalabteilung wünscht eine Abfrage, um für jeden Mitarbeiter Nachname, Tätigkeits-ID, Einstellungsdatum und Personalnummer anzuzeigen. Dabei soll die Personalnummer als erster Wert angezeigt werden. Geben Sie für die Spalte `HIRE_DATE` den Alias `STARTDATE` an. Speichern Sie die SQL-Anweisung als `lab_02_05b.sql`, damit Sie diese Datei an die Personalabteilung weiterleiten können. Testen Sie die Abfrage in der Datei `lab_02_05b.sql`, um sicherzustellen, dass sie korrekt ausgeführt wird.

**Hinweis:** Nachdem Sie die Abfrage ausgeführt haben, achten Sie darauf, die nächste Abfrage nicht in das gleiche Worksheet einzugeben, sondern ein neues Worksheet zu öffnen.

	 EMPLOYEE_ID	 LAST_NAME	 JOB_ID	 STARTDATE
1	100	King	AD_PRES	17-JUN-03
2	101	Kochhar	AD_VP	21-SEP-05
3	102	De Haan	AD_VP	13-JAN-01
4	103	Hunold	AC_MGR	03-JAN-06
5	104	Ernst	IT_PROG	21-MAY-07
6	107	Lorentz	IT_PROG	07-FEB-07
7	124	Mourgos	ST_MAN	16-NOV-07
8	141	Rajs	ST_CLERK	17-OCT-03
9	142	Davies	ST_CLERK	29-JAN-05
10	143	Matos	ST_CLERK	15-MAR-06
11	144	Vargas	ST_CLERK	09-JUL-06
12	149	Zlotkey	SA_MAN	29-JAN-08
13	174	Abel	SA_REP	11-MAY-04
14	176	Taylor	SA_REP	24-MAR-06
15	178	Grant	SA_REP	24-MAY-07
16	200	Whalen	AD_ASST	17-SEP-03
17	201	Hartstein	MK_MAN	17-FEB-04
18	202	Fay	MK_REP	17-AUG-05
19	205	Higgins	AC_MGR	07-JUN-02
20	206	Gietz	AC_ACCOUNT	07-JUN-02

6. Die Personalabteilung benötigt eine Abfrage, um alle eindeutigen Tätigkeits-IDs aus der Tabelle EMPLOYEES anzuzeigen.

 JOB_ID
1 AC_ACCOUNT
2 AC_MGR
3 AD_ASST
4 AD_PRES
5 AD_VP
6 IT_PROG
7 MK_MAN
8 MK_REP
9 SA_MAN
10 SA_REP
11 ST_CLERK
12 ST_MAN

### 3. Aufgabe

Führen Sie die folgenden Übungen durch, falls Sie noch Zeit haben:

- Die Personalabteilung wünscht aussagekräftigere Spaltenüberschriften für ihre Mitarbeiterberichte. Kopieren Sie die Anweisung aus lab\_02\_05b.sql in ein neues SQL Worksheet. Nennen Sie die Spalten Emp #, Employee, Job bzw. Hire Date. Führen Sie die Abfrage anschließend erneut aus.

	Emp #	Employee	Job	Hire Date
1	100	King	AD_PRES	17-JUN-03
2	101	Kochhar	AD_VP	21-SEP-05
3	102	De Haan	AD_VP	13-JAN-01
4	103	Hunold	AC_MGR	03-JAN-06
5	104	Ernst	IT_PROG	21-MAY-07
6	107	Lorentz	IT_PROG	07-FEB-07
7	124	Mourgos	ST_MAN	16-NOV-07
8	141	Rajs	ST_CLERK	17-OCT-03
9	142	Davies	ST_CLERK	29-JAN-05
10	143	Matos	ST_CLERK	15-MAR-06
11	144	Vargas	ST_CLERK	09-JUL-06
12	149	Zlotkey	SA_MAN	29-JAN-08
13	174	Abel	SA_REP	11-MAY-04
14	176	Taylor	SA_REP	24-MAR-06
15	178	Grant	SA_REP	24-MAY-07
16	200	Whalen	AD_ASST	17-SEP-03
17	201	Hartstein	MK_MAN	17-FEB-04
18	202	Fay	MK_REP	17-AUG-05
19	205	Higgins	AC_MGR	07-JUN-02
20	206	Gietz	AC_ACCOUNT	07-JUN-02

- Die Personalabteilung hat einen Bericht über alle Mitarbeiter und deren Tätigkeits-IDs angefordert. Verketteten Sie den Nachnamen mit der Tätigkeits-ID (durch Komma und Leerzeichen getrennt), und nennen Sie die Spalte Employee and Title.

	Employee and Title
1	Abel, SA_REP
2	Davies, ST_CLERK
3	De Haan, AD_VP
4	Ernst, IT_PROG
5	Fay, MK_REP
6	Gietz, AC_ACCOUNT

...

19	Whalen, AD_ASST
20	Zlotkey, SA_MAN

Wenn Sie eine weitere Herausforderung suchen, bearbeiten Sie die folgende Aufgabe:

9. Um sich mit der Tabelle `EMPLOYEES` vertraut zu machen, erstellen Sie eine Abfrage, die alle Daten aus dieser Tabelle anzeigt. Trennen Sie die Spaltenausgabe durch Kommas. Nennen Sie die Spalte `THE_OUTPUT`.

THE_OUTPUT
1 100,Steven,King,SKING,515.123.4567,AD_PRES,,17-JUN-03,24000,,90
2 101,Neena,Kochhar,NK0CHHAR,515.123.4568,AD_VP,100,21-SEP-05,17000,,90
3 102,Lex,De Haan,LDEHAAN,515.123.4569,AD_VP,100,13-JAN-01,17000,,90
4 103,Alexander,Hunold,AHUNOLD,590.423.4567,AC_MGR,102,03-JAN-06,12008,,60
5 104,Bruce,Ernst,BERNST,590.423.4568,IT_PROG,103,21-MAY-07,6000,,60
6 107,Diana,Lorentz,DLORENTZ,590.423.5567,IT_PROG,103,07-FEB-07,4200,,60

...

18 202,Pat,Fay,PFAY,603.123.6666,MK_REP,201,17-AUG-05,6000,,20
19 205,Shelley,Higgins,SHIGGINS,515.123.8080,AC_MGR,101,07-JUN-02,12008,,110
20 206,William,Gietz,WGIETZ,515.123.8181,AC_ACCOUNT,205,07-JUN-02,8300,,110

## Übung 1 zu Lektion 2 – Lösung: Daten mit der SQL-Anweisung SELECT abrufen

---

### 1. Aufgabe

Testen Sie Ihre Kenntnisse:

1. Die folgende Anweisung SELECT wird erfolgreich ausgeführt:

```
SELECT last_name, job_id, salary AS Sal
FROM employees;
```

**Richtig/Falsch**

2. Die folgende Anweisung SELECT wird erfolgreich ausgeführt:

```
SELECT *
FROM job_grades;
```

**Richtig/Falsch**

3. Die folgende Anweisung enthält vier Codierungsfehler. Können Sie alle vier Fehler finden?

```
SELECT      employee_id, last_name
sal x 12    ANNUAL SALARY
FROM        employees;
```

- **Die Tabelle EMPLOYEES enthält keine Spalte sal. Der richtige Spaltenname lautet SALARY.**
- **Der Multiplikationsoperator ist \*, nicht x, wie in Zeile 2 angegeben.**
- **Der Alias ANNUAL SALARY darf keine Leerzeichen enthalten. Der Alias muss ANNUAL\_SALARY lauten oder in doppelte Anführungszeichen gesetzt werden.**
- **Hinter dem Spaltennamen LAST\_NAME fehlt ein Komma.**

### 2. Aufgabe

Sie wurden als SQL-Programmierer bei Acme eingestellt. Ihre erste Aufgabe besteht darin, einige Berichte auf Basis der Daten aus den Human Resources-Tabellen zu erstellen.

4. Prüfen Sie zunächst Struktur und Inhalt der Tabelle DEPARTMENTS.

- a. So bestimmen Sie die Struktur der Tabelle DEPARTMENTS:

```
DESCRIBE departments
```

- b. So zeigen Sie die Daten der Tabelle DEPARTMENTS an:

```
SELECT *
FROM departments;
```

5. Prüfen Sie Struktur und Inhalt der Tabelle EMPLOYEES.

- a. Bestimmen Sie die Struktur der Tabelle EMPLOYEES.

```
DESCRIBE employees
```

- b. Die Personalabteilung wünscht eine Abfrage, um für jeden Mitarbeiter Nachname, Tätigkeits-ID, Einstellungsdatum und Personalnummer anzuzeigen. Dabei soll die Personalnummer als erster Wert angezeigt werden. Geben Sie für die Spalte `HIRE_DATE` den Alias `STARTDATE` an. Speichern Sie die SQL-Anweisung als `lab_02_05b.sql`, damit Sie diese Datei an die Personalabteilung weiterleiten können. Testen Sie die Abfrage in der Datei `lab_02_05b.sql`, um sicherzustellen, dass sie korrekt ausgeführt wird.

```
SELECT employee_id, last_name, job_id, hire_date StartDate
FROM employees;
```

6. Die Personalabteilung benötigt eine Abfrage, um alle eindeutigen Tätigkeits-IDs aus der Tabelle `EMPLOYEES` anzuzeigen.

```
SELECT DISTINCT job_id
FROM employees;
```

### 3. Aufgabe

Führen Sie die folgenden Übungen durch, falls Sie noch Zeit haben:

7. Die Personalabteilung wünscht aussagekräftigere Spaltenüberschriften für ihre Mitarbeiterberichte. Kopieren Sie die Anweisung aus `lab_02_05b.sql` in ein neues SQL Worksheet. Nennen Sie die Spalten `Emp #`, `Employee`, `Job` bzw. `Hire Date`. Führen Sie die Abfrage anschließend erneut aus.

```
SELECT employee_id "Emp #", last_name "Employee",
       job_id "Job", hire_date "Hire Date"
FROM employees;
```

8. Die Personalabteilung hat einen Bericht über alle Mitarbeiter und deren Tätigkeits-IDs angefordert. Verketteten Sie den Nachnamen mit der Tätigkeits-ID (durch Komma und Leerzeichen getrennt), und nennen Sie die Spalte `Employee and Title`.

```
SELECT last_name || ', ' || job_id "Employee and Title"
FROM employees;
```

Wenn Sie eine weitere Herausforderung suchen, bearbeiten Sie die folgende Aufgabe:

9. Um sich mit der Tabelle `EMPLOYEES` vertraut zu machen, erstellen Sie eine Abfrage, die alle Daten aus dieser Tabelle anzeigt. Trennen Sie die Spaltenausgabe durch Kommas. Nennen Sie die Spalte `THE_OUTPUT`.

```
SELECT employee_id || ', ' || first_name || ', ' || last_name
       || ', ' || email || ', ' || phone_number || ', ' || job_id
       || ', ' || manager_id || ', ' || hire_date || ', '
       || salary || ', ' || commission_pct || ', ' ||
       department_id
       THE_OUTPUT
FROM employees;
```





# **Übungen zu Lektion 3 – Daten einschränken und sortieren**

## **Kapitel 3**

# Übungen zu Lektion 3 – Überblick

---

## Übungsüberblick

Diese Übungen behandeln folgende Themen:

- Daten wählen und Reihenfolge der angezeigten Zeilen ändern
- Zeilen mit der Klausel `WHERE` einschränken
- Zeilen mit der Klausel `ORDER BY` sortieren
- `SELECT`-Anweisungen durch Substitutionsvariablen flexibler gestalten

## Übung 1 zu Lektion 3 – Daten einschränken und sortieren

---

### Überblick

In dieser Übung erstellen Sie weitere Berichte mit Anweisungen, die `WHERE`- und `ORDER BY`-Klauseln enthalten. Mithilfe von Substitutionsvariablen gestalten Sie SQL-Anweisungen allgemeiner und erleichtern ihre Wiederverwendung.

### Aufgabe

Die Personalabteilung benötigt Ihre Unterstützung bei der Erstellung einiger Abfragen.

1. Aus Budgetgründen benötigt die Personalabteilung einen Bericht, der Nachname und Gehalt aller Mitarbeiter anzeigt, die mehr als \$ 12.000 verdienen. Speichern Sie die SQL-Anweisung unter dem Dateinamen `lab_03_01.sql`. Führen Sie die Abfrage aus.

	LAST_NAME	SALARY
1	King	24000
2	Kochhar	17000
3	De Haan	17000
4	Hartstein	13000
5	Higgins	12008

2. Öffnen Sie ein neues SQL Worksheet. Erstellen Sie einen Bericht, um Nachname und Abteilungsnummer für den Mitarbeiter mit der Personalnummer 176 anzuzeigen. Führen Sie die Abfrage aus.

	LAST_NAME	DEPARTMENT_ID
1	Taylor	80

3. Die Personalabteilung möchte Mitarbeiter mit einem besonders hohen bzw. niedrigen Gehalt ermitteln. Ändern Sie die Datei `lab_03_01.sql` so ab, dass Nachname und Gehalt aller Mitarbeiter angezeigt werden, deren Gehalt nicht im Bereich zwischen \$ 5.000 und \$ 12.000 liegt. Speichern Sie die SQL-Anweisung als `lab_03_03.sql`.

	LAST_NAME	SALARY
1	King	24000
2	Kochhar	17000
3	De Haan	17000
4	Lorentz	4200
5	Rajs	3500
6	Davies	3100
7	Matos	2600
8	Vargas	2500
9	Whalen	4400
10	Hartstein	13000
11	Higgins	12008

4. Erstellen Sie einen Bericht, um Nachname, Tätigkeits-ID und Einstellungsdatum der Mitarbeiter mit den Nachnamen Matos und Taylor anzuzeigen. Sortieren Sie die Abfrage in aufsteigender Reihenfolge nach Einstellungsdatum.

	LAST_NAME	JOB_ID	HIRE_DATE
1	Matos	ST_CLERK	15-MAR-06
2	Taylor	SA_REP	24-MAR-06

5. Zeigen Sie Nachname und Abteilungsnummer aller Mitarbeiter der Abteilungen 20 und 50 an, alphabetisch in aufsteigender Reihenfolge nach Nachname sortiert.

	LAST_NAME	DEPARTMENT_ID
1	Davies	50
2	Fay	20
3	Hartstein	20
4	Matos	50
5	Mourgos	50
6	Rajs	50
7	Vargas	50

6. Ändern Sie die Datei lab\_03\_03.sql so ab, dass Nachname und Gehalt der Mitarbeiter angezeigt werden, die zwischen \$ 5.000 und \$ 12.000 verdienen und in den Abteilungen 20 bzw. 50 arbeiten. Nennen Sie die Spalten Employee und Monthly Salary. Speichern Sie die Datei lab\_03\_03.sql als lab\_03\_06.sql. Führen Sie die Anweisung in lab\_03\_06.sql aus.

	Employee	Monthly Salary
1	Fay	6000
2	Mourgos	5800

7. Die Personalabteilung benötigt einen Bericht, der Nachname und Einstellungsdatum für alle im Jahr 2006 eingestellten Mitarbeiter anzeigt.

	LAST_NAME	HIRE_DATE
1	Hunold	03-JAN-06
2	Matos	15-MAR-06
3	Vargas	09-JUL-06
4	Taylor	24-MAR-06

8. Erstellen Sie einen Bericht, um Nachname und Tätigkeits-ID aller Mitarbeiter anzuzeigen, denen kein Manager zugewiesen ist.

	LAST_NAME	JOB_ID
1	King	AD_PRES

9. Erstellen Sie einen Bericht, um Nachname, Gehalt und Provision aller provisionsberechtigten Mitarbeiter anzuzeigen. Sortieren Sie die Daten in absteigender Reihenfolge nach Gehalt und Provision.

Verwenden Sie in der Klausel `ORDER BY` die numerische Position der Spalte.

	LAST_NAME	SALARY	COMMISSION_PCT
1	Abel	11000	0.3
2	Zlotkey	10500	0.2
3	Taylor	8600	0.2
4	Grant	7000	0.15

10. Die Mitarbeiter der Personalabteilung wünschen mehr Flexibilität bei den von Ihnen erstellten Abfragen. Sie hätten gerne einen Bericht, der Nachname und Gehalt aller Mitarbeiter anzeigt, deren Gehalt über einem bestimmten Betrag liegt, den der Benutzer nach einer Eingabeaufforderung angibt. Speichern Sie diese Abfrage unter dem Dateinamen `lab_03_10.sql`. (Sie können die Abfrage aus der ersten Aufgabe verwenden und entsprechend abändern.) Wenn Sie in der Eingabeaufforderung "12000" eingeben, zeigt der Bericht folgende Ergebnisse an:

	LAST_NAME	SALARY
1	King	24000
2	Kochhar	17000
3	De Haan	17000
4	Hartstein	13000
5	Higgins	12008

11. Die Personalabteilung möchte Berichte auf Basis eines Managers erstellen. Erstellen Sie eine Abfrage, die den Benutzer zur Eingabe einer Manager-ID auffordert und dann Personalnummer, Nachname, Gehalt und Abteilung aller diesem Manager zugeordneten Mitarbeiter ausgibt. Die Personalabteilung möchte den Bericht nach einer beliebigen Spalte sortieren können. Sie können die Daten mit folgenden Werten testen:

`manager_id = 103`, sortiert nach `last_name`:

	EMPLOYEE_ID	LAST_NAME	SALARY	DEPARTMENT_ID
1	104	Ernst	6000	60
2	107	Lorentz	4200	60

`manager_id = 201`, sortiert nach `salary`:

	EMPLOYEE_ID	LAST_NAME	SALARY	DEPARTMENT_ID
1	202	Fay	6000	20

`manager_id = 124`, sortiert nach `employee_id`:

	EMPLOYEE_ID	LAST_NAME	SALARY	DEPARTMENT_ID
1	141	Rajs	3500	50
2	142	Davies	3100	50
3	143	Matos	2600	50
4	144	Vargas	2500	50

Führen Sie die folgenden Übungen durch, falls Sie noch Zeit haben:

12. Zeigen Sie die Nachnamen aller Mitarbeiter an, deren Name an dritter Stelle den Buchstaben "a" enthält.

	LAST_NAME
1	Grant
2	Whalen

13. Zeigen Sie die Nachnamen aller Mitarbeiter an, in deren Nachname ein "a" und ein "e" vorkommt.

	LAST_NAME
1	Davies
2	De Haan
3	Hartstein
4	Whalen

Wenn Sie eine weitere Herausforderung suchen, bearbeiten Sie folgende Aufgaben:

14. Zeigen Sie Nachname, Tätigkeits-ID und Gehalt aller Mitarbeiter an, die Sales Representative oder Stock Clerk sind und deren Gehalt nicht \$ 2.500, \$ 3.500 oder \$ 7.000 beträgt.

	LAST_NAME	JOB_ID	SALARY
1	Abel	SA_REP	11000
2	Taylor	SA_REP	8600
3	Davies	ST_CLERK	3100
4	Matos	ST_CLERK	2600

15. Ändern Sie die Datei lab\_03\_06.sql so ab, dass Nachname, Gehalt und Provision aller Mitarbeiter angezeigt werden, deren Provision 20 % beträgt. Speichern Sie die Datei lab\_03\_06.sql als lab\_03\_15.sql. Führen Sie die Anweisung in lab\_03\_15.sql erneut aus.

	Employee	Monthly Salary	COMMISSION_PCT
1	Zlotkey	10500	0.2
2	Taylor	8600	0.2

## Übung 1 zu Lektion 3 – Lösung: Daten einschränken und sortieren

---

Die Personalabteilung benötigt Ihre Unterstützung bei der Erstellung einiger Abfragen.

1. Aus Budgetgründen benötigt die Personalabteilung einen Bericht, der Nachname und Gehalt aller Mitarbeiter anzeigt, die mehr als \$ 12.000 verdienen. Speichern Sie die SQL-Anweisung unter dem Dateinamen `lab_03_01.sql`. Führen Sie die Abfrage aus.

```
SELECT last_name, salary
FROM employees
WHERE salary > 12000;
```

2. Öffnen Sie ein neues SQL Worksheet. Erstellen Sie einen Bericht, um Nachname und Abteilungsnummer für den Mitarbeiter mit der Personalnummer 176 anzuzeigen.

```
SELECT last_name, department_id
FROM employees
WHERE employee_id = 176;
```

3. Die Personalabteilung möchte Mitarbeiter mit einem besonders hohen bzw. niedrigen Gehalt ermitteln. Ändern Sie die Datei `lab_03_01.sql` so ab, dass Nachname und Gehalt aller Mitarbeiter angezeigt werden, deren Gehalt nicht im Bereich zwischen \$ 5.000 und \$ 12.000 liegt. Speichern Sie die SQL-Anweisung als `lab_03_03.sql`.

```
SELECT last_name, salary
FROM employees
WHERE salary NOT BETWEEN 5000 AND 12000;
```

4. Erstellen Sie einen Bericht, um Nachname, Tätigkeits-ID und Einstellungsdatum der Mitarbeiter mit den Nachnamen Matos und Taylor anzuzeigen. Sortieren Sie die Abfrage in aufsteigender Reihenfolge nach Einstellungsdatum.

```
SELECT last_name, job_id, hire_date
FROM employees
WHERE last_name IN ('Matos', 'Taylor')
ORDER BY hire_date;
```

5. Zeigen Sie Nachname und Abteilungsnummer aller Mitarbeiter der Abteilungen 20 und 50 an, und zwar alphabetisch in aufsteigender Reihenfolge nach Nachname sortiert.

```
SELECT last_name, department_id
FROM employees
WHERE department_id IN (20, 50)
ORDER BY last_name ASC;
```

6. Ändern Sie die Datei `lab_03_03.sql` so ab, dass Nachname und Gehalt der Mitarbeiter aufgeführt werden, die zwischen \$ 5.000 und \$ 12.000 verdienen und in den Abteilungen 20 bzw. 50 arbeiten. Nennen Sie die Spalten `Employee` und `Monthly Salary`. Speichern Sie die Datei `lab_03_03.sql` als `lab_03_06.sql`. Führen Sie die Anweisung in `lab_03_06.sql` aus.

```
SELECT    last_name "Employee", salary "Monthly Salary"
FROM      employees
WHERE     salary BETWEEN 5000 AND 12000
AND       department_id IN (20, 50);
```

7. Die Personalabteilung benötigt einen Bericht, der Nachname und Einstellungsdatum für alle im Jahr 2006 eingestellten Mitarbeiter anzeigt.

```
SELECT    last_name, hire_date
FROM      employees
WHERE     hire_date >= '01-JAN-06' AND hire_date < '01-JAN-07';
```

8. Erstellen Sie einen Bericht, um Nachname und Tätigkeits-ID aller Mitarbeiter anzuzeigen, denen kein Manager zugewiesen ist.

```
SELECT    last_name, job_id
FROM      employees
WHERE     manager_id IS NULL;
```

9. Erstellen Sie einen Bericht, um Nachname, Gehalt und Provision aller provisionsberechtigten Mitarbeiter anzuzeigen. Sortieren Sie die Daten in absteigender Reihenfolge nach Gehalt und Provision. Verwenden Sie in der Klausel `ORDER BY` die numerische Position der Spalte.

```
SELECT    last_name, salary, commission_pct
FROM      employees
WHERE     commission_pct IS NOT NULL
ORDER BY 2 DESC, 3 DESC;
```

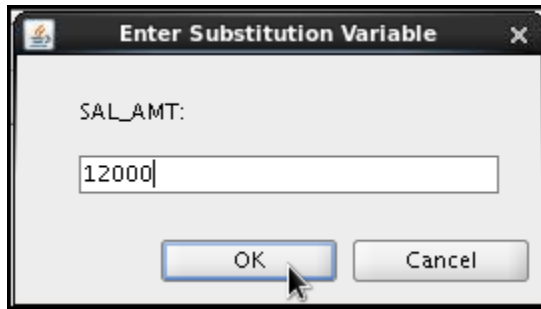
10. Die Mitarbeiter der Personalabteilung wünschen mehr Flexibilität bei den von Ihnen erstellten Abfragen. Sie hätten gerne einen Bericht, der Nachname und Gehalt aller Mitarbeiter anzeigt, deren Gehalt über einem bestimmten Betrag liegt, den der Benutzer nach einer Eingabeaufforderung angibt. (Sie können die Abfrage aus der ersten Aufgabe verwenden und entsprechend abändern.) Speichern Sie diese Abfrage unter dem Dateinamen `lab_03_10.sql`.

Wenn Sie zur Eingabe aufgefordert werden, geben Sie 12000 ein:

```
SELECT    last_name, salary
FROM      employees
WHERE     salary > &sal_amt;
```



Wenn Sie im Dialogfeld zur Eingabe eines Wertes aufgefordert werden, geben Sie 12000 ein. Klicken Sie auf **OK**.



11. Die Personalabteilung möchte Berichte auf der Basis eines Managers erstellen. Erstellen Sie eine Abfrage, die den Benutzer zur Eingabe einer Manager-ID auffordert und dann Personalnummer, Nachname, Gehalt und Abteilung aller diesem Manager zugeordneten Mitarbeiter ausgibt. Die Personalabteilung möchte den Bericht nach einer beliebigen Spalte sortieren können. Sie können die Daten mit folgenden Werten testen:

manager\_id = 103, sortiert nach last\_name

manager\_id = 201, sortiert nach salary

manager\_id = 124, sortiert nach employee\_id

```
SELECT employee_id, last_name, salary, department_id
FROM employees
WHERE manager_id = &mgr_num
ORDER BY &order_col;
```

Führen Sie die folgenden Übungen durch, falls Sie noch Zeit haben:

12. Zeigen Sie die Nachnamen aller Mitarbeiter an, deren Name an dritter Stelle den Buchstaben "a" enthält.

```
SELECT last_name
FROM employees
WHERE last_name LIKE '__a%';
```

13. Zeigen Sie die Nachnamen aller Mitarbeiter an, in deren Nachname ein "a" und ein "e" vorkommt.

```
SELECT last_name
FROM employees
WHERE last_name LIKE '%a%'
AND last_name LIKE '%e%';
```

Wenn Sie eine weitere Herausforderung suchen, bearbeiten Sie folgende Aufgaben:

14. Zeigen Sie Nachname, Tätigkeits-ID und Gehalt aller Mitarbeiter an, die Sales Representative oder Stock Clerk sind und deren Gehalt nicht \$ 2.500, \$ 3.500 oder \$ 7.000 beträgt.

```
SELECT last_name, job_id, salary
FROM employees
WHERE job_id IN ('SA_REP', 'ST_CLERK')
AND salary NOT IN (2500, 3500, 7000);
```

15. Ändern Sie die Datei lab\_03\_06.sql so ab, dass Nachname, Gehalt und Provision aller Mitarbeiter angezeigt werden, deren Provision 20 % beträgt. Speichern Sie die Datei lab\_03\_06.sql als lab\_03\_15.sql. Führen Sie die Anweisung in lab\_03\_15.sql erneut aus.

```
SELECT last_name "Employee", salary "Monthly Salary",
       commission_pct
FROM employees
WHERE commission_pct = .20;
```

# **Übungen zu Lektion 4 – Ausgabe mit Single-Row- Funktionen anpassen**

## **Kapitel 4**

## Übungen zu Lektion 4 – Überblick

---

### Übungsüberblick

Diese Übung behandelt folgende Themen:

- Abfrage erstellen, die das aktuelle Datum anzeigt
- Abfragen mithilfe von numerischen Funktionen, Zeichenfunktionen und Datumsfunktionen erstellen
- Beschäftigungsdauer eines Mitarbeiters in Jahren und Monaten berechnen

# Übung 1 zu Lektion 4 – Ausgabe mit Single-Row-Funktionen anpassen

## Überblick

Diese Übung enthält eine Reihe von Aufgaben, in denen die verschiedenen Funktionen verwendet werden, die für die Datentypen CHARACTER, NUMBER und DATE verfügbar sind. Beachten Sie, dass verschachtelte Funktionen von innen nach außen ausgewertet werden.

## Aufgaben

1. Erstellen Sie eine Abfrage, um das Systemdatum anzuzeigen. Nennen Sie die Spalte `Date`.

**Hinweis:** Wenn Sie eine Remote-Datenbank in einer anderen Zeitzone verwenden, zeigt die Ausgabe das Datum des Betriebssystems an, auf dem sich die Datenbank befindet.

Date
1 30-AUG-12

2. Die Personalabteilung benötigt einen Bericht, um für jeden Mitarbeiter die Personalnummer, den Nachnamen, das Gehalt und eine Gehaltserhöhung um 15,5 % (als ganze Zahl ausgedrückt) anzuzeigen. Nennen Sie die Spalte `New Salary`. Speichern Sie Ihre SQL-Anweisung in der Datei `lab_04_02.sql`.
3. Führen Sie die Abfrage in der Datei `lab_04_02.sql` aus.

	EMPLOYEE_ID	LAST_NAME	SALARY	New Salary
1	100	King	24000	27720
2	101	Kochhar	17000	19635
3	102	De Haan	17000	19635
4	103	Hunold	9000	10395
5	104	Ernst	6000	6930
6	107	Lorentz	4200	4851
7	124	Mourgos	5800	6699
8	141	Rajs	3500	4043
9	142	Davies	3100	3581
10	143	Matos	2600	3003
11	144	Vargas	2500	2888
12	149	Zlotkey	10500	12128
13	174	Abel	11000	12705
14	176	Taylor	8600	9933
15	178	Grant	7000	8085
16	200	Whalen	4400	5082
17	201	Hartstein	13000	15015
18	202	Fay	6000	6930
19	205	Higgins	12008	13869
20	206	Gietz	8300	9587

4. Ändern Sie die Abfrage aus `lab_04_02.sql`, indem Sie eine Spalte hinzufügen, in der das alte Gehalt vom neuen Gehalt subtrahiert wird. Nennen Sie die Spalte `Increase`. Speichern Sie den Inhalt der Datei in `lab_04_04.sql`. Führen Sie die geänderte Abfrage aus.

	EMPLOYEE_ID	LAST_NAME	SALARY	New Salary	Increase
1	100	King	24000	27720	3720
2	101	Kochhar	17000	19635	2635
3	102	De Haan	17000	19635	2635
4	103	Hunold	9000	10395	1395
5	104	Ernst	6000	6930	930
6	107	Lorentz	4200	4851	651
7	124	Mourgos	5800	6699	899
8	141	Rajs	3500	4043	543
9	142	Davies	3100	3581	481
10	143	Matos	2600	3003	403
11	144	Vargas	2500	2888	388
12	149	Zlotkey	10500	12128	1628
13	174	Abel	11000	12705	1705
14	176	Taylor	8600	9933	1333
15	178	Grant	7000	8085	1085
16	200	Whalen	4400	5082	682
17	201	Hartstein	13000	15015	2015
18	202	Fay	6000	6930	930
19	205	Higgins	12008	13869	1861
20	206	Gietz	8300	9587	1287

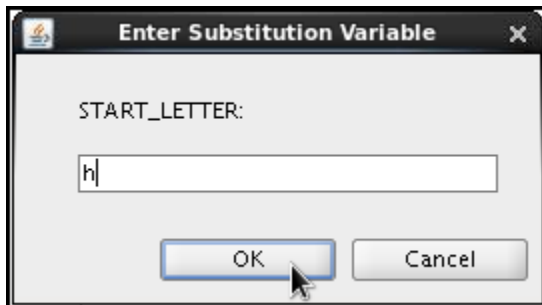
5. Führen Sie folgende Aufgaben aus:
- a. Erstellen Sie eine Abfrage, um für alle Mitarbeiter, deren Name mit "J", "A" oder "M" beginnt, den Nachnamen (erster Buchstabe in Groß- und alle anderen Buchstaben in Kleinschreibung) und die Länge des Nachnamens anzuzeigen. Benennen Sie die einzelnen Spalten entsprechend. Sortieren Sie die Ergebnisse nach den Nachnamen der Mitarbeiter.

	Name	Length
1	Abel	4
2	Matos	5
3	Mourgos	7

- b. Schreiben Sie die Abfrage so um, dass der Benutzer zur Eingabe des ersten Buchstabens des Nachnamens aufgefordert wird. Beispiel: Wenn der Benutzer ein "H" (Großbuchstabe) eingibt, sollen in der Ausgabe alle Mitarbeiter angezeigt werden, deren Nachname mit dem Buchstaben "H" beginnt.

	Name	Length
1	Hartstein	9
2	Higgins	7
3	Hunold	6

- c. Ändern Sie die Abfrage so, dass sich die Schreibweise bei der Eingabe (Groß- oder Kleinbuchstabe) nicht auf die Ausgabe auswirkt. Der eingegebene Buchstabe muss in Großschreibung umgewandelt werden, bevor er von der Abfrage `SELECT` verarbeitet wird.



	Name	Length
1	Hartstein	9
2	Higgins	7
3	Hunold	6

Führen Sie die folgenden Übungen durch, falls Sie noch Zeit haben:

- Die Personalabteilung möchte die Beschäftigungsdauer für alle Mitarbeiter ermitteln. Zeigen Sie für jeden Mitarbeiter den Nachnamen an, und berechnen Sie die Anzahl der Monate zwischen dem aktuellen Datum und dem Einstellungsdatum. Nennen Sie die Spalte `MONTHS_WORKED`. Sortieren Sie die Ergebnisse nach der Beschäftigungsdauer. Die Anzahl der Monate muss auf die nächstliegende Ganzzahl gerundet werden.

**Hinweis:** Da diese Abfrage vom Datum ihrer Ausführung abhängt, stimmen die bei Ihnen in der Spalte `MONTHS_WORKED` angezeigten Werte nicht mit den hier gezeigten Angaben überein.

R	LAST_NAME	R	MONTHS_WORKED
1	Zlotkey		55
2	Mourgos		57
3	Grant		63
4	Ernst		63
5	Lorentz		67
6	Vargas		74
7	Matos		77
8	Taylor		77
9	Hunold		80
10	Kochhar		83
11	Fay		84
12	Davies		91
13	Abel		100
14	Hartstein		102
15	Rajs		106
16	Whalen		107
17	King		110
18	Higgins		123
19	Gietz		123
20	De Haan		140



7. Erstellen Sie eine Abfrage, um Nachname und Gehalt aller Mitarbeiter anzuzeigen. Formatieren Sie die Gehaltsangaben wie folgt: 15 Zeichen lang und linksbündig mit dem Symbol \$ aufgefüllt. Nennen Sie die Spalte `SALARY`.

	LAST_NAME	SALARY
1	King	\$\$\$\$\$\$\$\$\$24000
2	Kochhar	\$\$\$\$\$\$\$\$\$17000
3	De Haan	\$\$\$\$\$\$\$\$\$17000
4	Hunold	\$\$\$\$\$\$\$\$\$9000
5	Ernst	\$\$\$\$\$\$\$\$\$6000
6	Lorentz	\$\$\$\$\$\$\$\$\$4200
7	Mourgos	\$\$\$\$\$\$\$\$\$5800
8	Rajs	\$\$\$\$\$\$\$\$\$3500
9	Davies	\$\$\$\$\$\$\$\$\$3100
10	Matos	\$\$\$\$\$\$\$\$\$2600
11	Vargas	\$\$\$\$\$\$\$\$\$2500
12	Zlotkey	\$\$\$\$\$\$\$\$\$10500
13	Abel	\$\$\$\$\$\$\$\$\$11000
14	Taylor	\$\$\$\$\$\$\$\$\$8600
15	Grant	\$\$\$\$\$\$\$\$\$7000
16	Whalen	\$\$\$\$\$\$\$\$\$4400
17	Hartstein	\$\$\$\$\$\$\$\$\$13000
18	Fay	\$\$\$\$\$\$\$\$\$6000
19	Higgins	\$\$\$\$\$\$\$\$\$12008
20	Gietz	\$\$\$\$\$\$\$\$\$8300

8. Erstellen Sie eine Abfrage, um den Nachnamen der Mitarbeiter anzuzeigen und den Betrag ihres Jahresgehalts durch Sternchen wiederzugeben. Jedes Sternchen steht für tausend Dollar. Sortieren Sie die Daten absteigend nach Gehalt. Nennen Sie die Spalte EMPLOYEES\_AND\_THEIR\_SALARIES.

	LAST_NAME	EMPLOYEES_AND_THEIR_SALARIES
1	King	*****
2	Kochhar	*****
3	De Haan	*****
4	Hartstein	*****
5	Higgins	*****
6	Abel	*****
7	Zlotkey	*****
8	Hunold	*****
9	Taylor	*****
10	Gietz	*****
11	Grant	*****
12	Ernst	*****
13	Fay	*****
14	Mourgos	*****
15	Whalen	*****
16	Lorentz	*****
17	Rajs	*****
18	Davies	*****
19	Matos	*****
20	Vargas	*****

9. Erstellen Sie eine Abfrage, um für alle Mitarbeiter aus Abteilung 90 Nachname und Beschäftigungsdauer (in Wochen) anzuzeigen. Nennen Sie die Spalte für die Wochenanzahl TENURE. Schneiden Sie den Wert der Wochenanzahl auf 0 Dezimalstellen ab. Zeigen Sie die Datensätze in absteigender Reihenfolge der Beschäftigungsdauer an.

**Hinweis:** Der Wert TENURE variiert, da er vom Datum abhängt, an dem Sie die Abfrage ausführen.

	LAST_NAME	TENURE
1	De Haan	606
2	King	480
3	Kochhar	362

## Übung 1 zu Lektion 4 – Lösung: Ausgabe mit Single-Row-Funktionen anpassen

---

1. Erstellen Sie eine Abfrage, um das Systemdatum anzuzeigen. Nennen Sie die Spalte `Date`.

**Hinweis:** Wenn Sie eine Remote-Datenbank in einer anderen Zeitzone verwenden, zeigt die Ausgabe das Datum des Betriebssystems an, auf dem sich die Datenbank befindet.

```
SELECT sysdate "Date"
FROM dual;
```

2. Die Personalabteilung benötigt einen Bericht, um für jeden Mitarbeiter die Personalnummer, den Nachnamen, das Gehalt und eine Gehaltserhöhung um 15,5 % (als ganze Zahl ausgedrückt) anzuzeigen. Nennen Sie die Spalte `New Salary`. Speichern Sie Ihre SQL-Anweisung in der Datei `lab_04_02.sql`.

```
SELECT employee_id, last_name, salary,
       ROUND(salary * 1.155, 0) "New Salary"
FROM employees;
```

3. Führen Sie die Abfrage in der Datei `lab_04_02.sql` aus.

```
SELECT employee_id, last_name, salary,
       ROUND(salary * 1.155, 0) "New Salary"
FROM employees;
```

4. Ändern Sie die Abfrage aus `lab_04_02.sql`, indem Sie eine Spalte hinzufügen, in der das alte Gehalt vom neuen Gehalt subtrahiert wird. Nennen Sie die Spalte `Increase`. Speichern Sie den Inhalt der Datei in `lab_04_04.sql`. Führen Sie die geänderte Abfrage aus.

```
SELECT employee_id, last_name, salary,
       ROUND(salary * 1.155, 0) "New Salary",
       ROUND(salary * 1.155, 0) - salary "Increase"
FROM employees;
```

5. Führen Sie folgende Aufgaben aus:

- a. Erstellen Sie eine Abfrage, um für alle Mitarbeiter, deren Name mit "J", "A" oder "M" beginnt, den Nachnamen (erster Buchstabe in Groß- und alle anderen Buchstaben in Kleinschreibung) und die Länge des Nachnamens anzuzeigen. Benennen Sie die einzelnen Spalten entsprechend. Sortieren Sie die Ergebnisse nach den Nachnamen der Mitarbeiter.

```
SELECT INITCAP(last_name) "Name",
       LENGTH(last_name) "Length"
FROM employees
WHERE last_name LIKE 'J%'
OR      last_name LIKE 'M%'
OR      last_name LIKE 'A%'
ORDER BY last_name;
```

- d. Schreiben Sie die Abfrage so um, dass der Benutzer aufgefordert wird, den ersten Buchstaben des Nachnamens einzugeben. Beispiel: Wenn der Benutzer ein H (Großbuchstabe) eingibt, sollen in der Ausgabe alle Mitarbeiter angezeigt werden, deren Nachname mit dem Buchstaben "H" beginnt.

```
SELECT  INITCAP(last_name) "Name",
        LENGTH(last_name) "Length"
FROM    employees
WHERE    last_name LIKE '&start_letter%'
ORDER BY last_name;
```

- e. Ändern Sie die Abfrage so, dass sich die Schreibweise bei der Eingabe (Groß- oder Kleinbuchstabe) nicht auf die Ausgabe auswirkt. Der eingegebene Buchstabe muss in Großschreibung umgewandelt werden, bevor er von der Abfrage SELECT verarbeitet wird.

```
SELECT  INITCAP(last_name) "Name",
        LENGTH(last_name) "Length"
FROM    employees
WHERE    last_name LIKE UPPER('&start_letter%' )
ORDER BY last_name;
```

Führen Sie die folgenden Übungen durch, falls Sie noch Zeit haben:

6. Die Personalabteilung möchte die Beschäftigungsdauer für alle Mitarbeiter ermitteln. Zeigen Sie für jeden Mitarbeiter den Nachnamen an, und berechnen Sie die Anzahl der Monate zwischen dem aktuellen Datum und dem Einstellungsdatum. Nennen Sie die Spalte MONTHS\_WORKED. Sortieren Sie die Ergebnisse nach der Beschäftigungsdauer. Die Anzahl der Monate muss auf die nächstliegende Ganzzahl gerundet werden.

**Hinweis:** Da diese Abfrage vom Datum ihrer Ausführung abhängt, stimmen Ihre Werte in der Spalte MONTHS\_WORKED nicht mit den hier gezeigten Angaben überein.

```
SELECT last_name, ROUND(MONTHS_BETWEEN(
        SYSDATE, hire_date)) MONTHS_WORKED
FROM    employees
ORDER BY months_worked;
```

7. Erstellen Sie eine Abfrage, um Nachname und Gehalt aller Mitarbeiter anzuzeigen. Formatieren Sie die Gehaltsangaben wie folgt: 15 Zeichen lang und linksbündig mit dem Symbol \$ aufgefüllt. Nennen Sie die Spalte SALARY.

```
SELECT last_name,
        LPAD(salary, 15, '$') SALARY
FROM    employees;
```

8. Erstellen Sie eine Abfrage, um den Nachnamen der Mitarbeiter anzuzeigen und den Betrag ihres Jahresgehalts durch Sternchen wiederzugeben. Jedes Sternchen steht für tausend Dollar. Sortieren Sie die Daten absteigend nach Gehalt. Nennen Sie die Spalte `EMPLOYEES_AND_THEIR_SALARIES`.

```
SELECT last_name,  
       rpad(' ', salary/1000, '*')  
         EMPLOYEES_AND_THEIR_SALARIES  
FROM   employees  
ORDER BY salary DESC;
```

9. Erstellen Sie eine Abfrage, um für alle Mitarbeiter aus Abteilung 90 Nachname und Beschäftigungsdauer (in Wochen) anzuzeigen. Nennen Sie die Spalte für die Wochenanzahl `TENURE`. Schneiden Sie den Wert der Wochenanzahl auf 0 Dezimalstellen ab. Zeigen Sie die Datensätze in absteigender Reihenfolge der Beschäftigungsdauer an.

**Hinweis:** Der Wert `TENURE` variiert, da er vom Datum abhängt, an dem Sie die Abfrage ausführen.

```
SELECT last_name, trunc((SYSDATE-hire_date)/7) AS TENURE  
FROM     employees  
WHERE    department_id = 90  
ORDER BY TENURE DESC;
```



# **Übungen zu Lektion 5 – Konvertierungsfunktionen und bedingte Ausdrücke**

## **Kapitel 5**

# Übungen zu Lektion 5 – Überblick

---

## Übungsüberblick

Diese Übung behandelt folgende Themen:

- Abfragen mit den Funktionen `TO_CHAR` und `TO_DATE` erstellen
- Abfragen mit bedingten Ausdrücken wie `CASE`, `SEARCHED CASE` und `DECODE` erstellen



## Übung 1 zu Lektion 5 – Konvertierungsfunktionen und bedingte Ausdrücke

---

### Überblick

Diese Übung enthält eine Reihe von Aufgaben zu den Funktionen TO\_CHAR und TO\_DATE sowie zu bedingten Ausdrücken wie CASE, Searched CASE und DECODE.

### Aufgaben

1. Erstellen Sie einen Bericht, der für jeden Mitarbeiter folgende Informationen enthält: <Nachname des Mitarbeiters> verdient monatlich <Gehalt>, aber fordert <3-faches Gehalt>. Nennen Sie die Spalte Dream Salaries.

	Dream Salaries
1	King earns \$24,000.00 monthly but wants \$72,000.00.
2	Kochhar earns \$17,000.00 monthly but wants \$51,000.00.
3	De Haan earns \$17,000.00 monthly but wants \$51,000.00.
4	Hunold earns \$12,008.00 monthly but wants \$36,024.00.
5	Ernst earns \$6,000.00 monthly but wants \$18,000.00.
6	Lorentz earns \$4,200.00 monthly but wants \$12,600.00.
7	Mourgos earns \$5,800.00 monthly but wants \$17,400.00.
8	Rajs earns \$3,500.00 monthly but wants \$10,500.00.
9	Davies earns \$3,100.00 monthly but wants \$9,300.00.
10	Matos earns \$2,600.00 monthly but wants \$7,800.00.
11	Vargas earns \$2,500.00 monthly but wants \$7,500.00.
12	Zlotkey earns \$10,500.00 monthly but wants \$31,500.00.
13	Abel earns \$11,000.00 monthly but wants \$33,000.00.
14	Taylor earns \$8,600.00 monthly but wants \$25,800.00.
15	Grant earns \$7,000.00 monthly but wants \$21,000.00.
16	Whalen earns \$4,400.00 monthly but wants \$13,200.00.
17	Hartstein earns \$13,000.00 monthly but wants \$39,000.00.
18	Fay earns \$6,000.00 monthly but wants \$18,000.00.
19	Higgins earns \$12,008.00 monthly but wants \$36,024.00.
20	Gietz earns \$8,300.00 monthly but wants \$24,900.00.

2. Zeigen Sie für jeden Mitarbeiter den Nachnamen, das Einstellungsdatum und das Datum der Gehaltsüberprüfung an (der erste Montag nach einer Beschäftigungsdauer von sechs Monaten). Nennen Sie die Spalte `REVIEW`. Formatieren Sie die Datumswerte in einem Format wie "Monday, the Thirty-First of July, 2000".

	LAST_NAME	HIRE_DATE	REVIEW
1	King	17-JUN-03	Monday, the Twenty-Second of December, 2003
2	Kochhar	21-SEP-05	Monday, the Twenty-Seventh of March, 2006
3	De Haan	13-JAN-01	Monday, the Sixteenth of July, 2001
4	Hunold	03-JAN-06	Monday, the Tenth of July, 2006
5	Ernst	21-MAY-07	Monday, the Twenty-Sixth of November, 2007
6	Lorentz	07-FEB-07	Monday, the Thirteenth of August, 2007
7	Mourgos	16-NOV-07	Monday, the Nineteenth of May, 2008
8	Rajs	17-OCT-03	Monday, the Nineteenth of April, 2004
9	Davies	29-JAN-05	Monday, the First of August, 2005
10	Matos	15-MAR-06	Monday, the Eighteenth of September, 2006
11	Vargas	09-JUL-06	Monday, the Fifteenth of January, 2007
12	Zlotkey	29-JAN-08	Monday, the Fourth of August, 2008
13	Abel	11-MAY-04	Monday, the Fifteenth of November, 2004
14	Taylor	24-MAR-06	Monday, the Twenty-Fifth of September, 2006
15	Grant	24-MAY-07	Monday, the Twenty-Sixth of November, 2007
16	Whalen	17-SEP-03	Monday, the Twenty-Second of March, 2004
17	Hartstein	17-FEB-04	Monday, the Twenty-Third of August, 2004
18	Fay	17-AUG-05	Monday, the Twentieth of February, 2006
19	Higgins	07-JUN-02	Monday, the Ninth of December, 2002
20	Gietz	07-JUN-02	Monday, the Ninth of December, 2002

3. Erstellen Sie eine Abfrage, die Nachname und Provisionsbetrag der Mitarbeiter anzeigt. Für nicht provisionsberechtigte Mitarbeiter soll "No Commission" angezeigt werden. Nennen Sie die Spalte COMM.

R	LAST_NAME	R	COMM
1	King		No Commission
2	Kochhar		No Commission
3	De Haan		No Commission
4	Hunold		No Commission
5	Ernst		No Commission
6	Lorentz		No Commission
7	Mourgos		No Commission
8	Rajs		No Commission
9	Davies		No Commission
10	Matos		No Commission
11	Vargas		No Commission
12	Zlotkey		.2
13	Abel		.3
14	Taylor		.2
15	Grant		.15
16	Whalen		No Commission
17	Hartstein		No Commission
18	Fay		No Commission
19	Higgins		No Commission
20	Gietz		No Commission

4. Erstellen Sie eine Abfrage mit der Funktion `CASE`, um die Gehaltsstufe eines jeden Mitarbeiters auf der Basis der Spalte `JOB_ID` anzuzeigen. Verwenden Sie hierbei folgende Daten:

<b>Tätigkeit</b>	<b>Gehaltsstufe</b>
AD_PRES	A
ST_MAN	B
IT_PROG	C
SA_REP	D
ST_CLERK	E
Keine der oben genannten	0

	JOB_ID	GRADE
1	AC_ACCOUNT	0
2	AC_MGR	0
3	AD_ASST	0
4	AD_PRES	A
5	AD_VP	0
6	AD_VP	0
7	IT_PROG	C
8	IT_PROG	C
9	IT_PROG	C
10	MK_MAN	0
11	MK_REP	0
12	SA_MAN	0
13	SA_REP	D
14	SA_REP	D
15	SA_REP	D
16	ST_CLERK	E
17	ST_CLERK	E
18	ST_CLERK	E
19	ST_CLERK	E
20	ST_MAN	B

5. Schreiben Sie die Anweisung aus der vorherigen Aufgabe mit der Searched CASE-Syntax um.

	JOB_ID	GRADE
1	AC_ACCOUNT	O
2	AC_MGR	O
3	AD_ASST	O
4	AD PRES	A
5	AD_VP	O
6	AD_VP	O
7	IT_PROG	C
8	IT_PROG	C
9	IT_PROG	C
10	MK_MAN	O
11	MK_REP	O
12	SA_MAN	O
13	SA_REP	D
14	SA_REP	D
15	SA_REP	D
16	ST_CLERK	E
17	ST_CLERK	E
18	ST_CLERK	E
19	ST_CLERK	E
20	ST_MAN	B

6. Schreiben Sie die Anweisung aus der vorherigen Aufgabe mit der `DECODE`-Syntax um.

	JOB_ID	GRADE
1	AC_ACCOUNT	O
2	AC_MGR	O
3	AD_ASST	O
4	AD PRES	A
5	AD_VP	O
6	AD_VP	O
7	IT_PROG	C
8	IT_PROG	C
9	IT_PROG	C
10	MK_MAN	O
11	MK_REP	O
12	SA_MAN	O
13	SA_REP	D
14	SA_REP	D
15	SA_REP	D
16	ST_CLERK	E
17	ST_CLERK	E
18	ST_CLERK	E
19	ST_CLERK	E
20	ST_MAN	B

## Übung 1 zu Lektion 5 – Lösung: Konvertierungsfunktionen und bedingte Ausdrücke

---

1. Erstellen Sie einen Bericht, der für jeden Mitarbeiter folgende Informationen enthält: <Nachname des Mitarbeiters> verdient monatlich <Gehalt>, aber fordert <3-faches Gehalt>. Nennen Sie die Spalte **Dream Salaries**.

```
SELECT last_name || ' earns '
       || TO_CHAR(salary, 'fm$99,999.00')
       || ' monthly but wants '
       || TO_CHAR(salary * 3, 'fm$99,999.00')
       || '.' "Dream Salaries"
FROM   employees;
```

2. Zeigen Sie für jeden Mitarbeiter den Nachnamen, das Einstellungsdatum und das Datum der Gehaltsüberprüfung an (der erste Montag nach einer Beschäftigungsdauer von sechs Monaten). Nennen Sie die Spalte **REVIEW**. Formatieren Sie die Datumswerte in einem Format wie "Monday, the Thirty-First of July, 2000".

```
SELECT last_name, hire_date,
       TO_CHAR(NEXT_DAY(ADD_MONTHS(hire_date, 6), 'MONDAY'),
       'fmDay, "the" Ddsph "of" Month, YYYY') REVIEW
FROM   employees;
```

3. Erstellen Sie eine Abfrage, die Nachname und Provisionsbetrag der Mitarbeiter anzeigt. Für nicht provisionsberechtigte Mitarbeiter soll "No Commission" angezeigt werden. Nennen Sie die Spalte **COMM**.

```
SELECT last_name,
       NVL(TO_CHAR(commission_pct), 'No Commission') COMM
FROM   employees;
```

4. Erstellen Sie eine Abfrage mit der Funktion **CASE**, um die Gehaltsstufe eines jeden Mitarbeiters auf der Basis der Spalte **JOB\_ID** anzuzeigen. Verwenden Sie hierbei folgende Daten:

<b>Tätigkeit</b>	<b>Gehaltsstufe</b>
AD_PRES	A
ST_MAN	B
IT_PROG	C
SA_REP	D
ST_CLERK	E
Keine der oben genannten	0

```

SELECT job_id, CASE job_id
                WHEN 'ST_CLERK' THEN 'E'
                WHEN 'SA_REP'   THEN 'D'
                WHEN 'IT_PROG'  THEN 'C'
                WHEN 'ST_MAN'   THEN 'B'
                WHEN 'AD_PRES'  THEN 'A'
                ELSE '0'   END  GRADE
FROM employees;

```

5. Schreiben Sie die Anweisung aus der vorherigen Aufgabe mit der Searched CASE-Syntax um.

```

SELECT job_id, CASE
                WHEN job_id = 'ST_CLERK' THEN 'E'
                WHEN job_id = 'SA_REP'   THEN 'D'
                WHEN job_id = 'IT_PROG'  THEN 'C'
                WHEN job_id = 'ST_MAN'   THEN 'B'
                WHEN job_id = 'AD_PRES'  THEN 'A'
                ELSE '0'   END  GRADE
FROM employees;

```

6. Schreiben Sie die Anweisung aus der vorherigen Aufgabe mit der DECODE-Syntax um.

```

SELECT job_id, decode (job_id,
                'ST_CLERK', 'E',
                'SA_REP',   'D',
                'IT_PROG',  'C',
                'ST_MAN',   'B',
                'AD_PRES',  'A',
                '0') GRADE
FROM employees;

```



# **Übungen zu Lektion 6 – Mit Gruppenfunktionen Berichte aus aggregierten Daten erstellen**

## **Kapitel 6**

## Übungen zu Lektion 6 – Überblick

---

### Übungsüberblick

Diese Übung behandelt folgende Themen:

- Abfragen mit Gruppenfunktionen erstellen
- Nach Zeilen gruppieren, um mehrere Ergebnisse zu erhalten
- Gruppen mit der Klausel `HAVING` einschränken

## Übung 1 zu Lektion 6 – Mit Gruppenfunktionen Berichte zu aggregierten Daten erstellen

### Überblick

Nach Abschluss dieser Übungen können Sie Gruppenfunktionen verwenden und Datengruppen wählen.

### Aufgaben

Prüfen Sie die Richtigkeit der folgenden Aussagen. Kreuzen Sie Richtig oder Falsch an.

1. Gruppenfunktionen bearbeiten viele Zeilen, um ein Ergebnis pro Gruppe auszugeben.  
Richtig/Falsch
2. Gruppenfunktionen berücksichtigen Nullwerte in den Berechnungen.  
Richtig/Falsch
3. Die Klausel `WHERE` schränkt die Zeilen vor der Aufnahme in eine Gruppenberechnung ein.  
Richtig/Falsch

Die Personalabteilung benötigt folgende Berichte:

4. Es sollen das höchste Gehalt, das niedrigste Gehalt, die Summe der Gehälter und das Durchschnittsgehalt für alle Mitarbeiter ermittelt werden. Nennen Sie die Spalten `Maximum`, `Minimum`, `Sum` und `Average`. Runden Sie die Ergebnisse auf die nächste Ganzzahl. Speichern Sie die SQL-Anweisung als `lab_06_04.sql`. Führen Sie die Abfrage aus.

	Maximum	Minimum	Sum	Average
1	24000	2500	175508	8775

5. Ändern Sie die Abfrage in der Datei `lab_06_04.sql`, um das niedrigste Gehalt, das höchste Gehalt, die Summe der Gehälter und das Durchschnittsgehalt für jede Tätigkeitsbezeichnung anzuzeigen. Speichern Sie die Datei `lab_06_04.sql` unter dem Namen `lab_06_05.sql`. Führen Sie die Anweisung in `lab_06_05.sql` aus.

	JOB_ID	Maximum	Minimum	Sum	Average
1	IT_PROG	9000	4200	19200	6400
2	AC_MGR	12008	12008	12008	12008
3	AC_ACCOUNT	8300	8300	8300	8300
4	ST_MAN	5800	5800	5800	5800
5	AD_ASST	4400	4400	4400	4400
6	AD_VP	17000	17000	34000	17000
7	SA_MAN	10500	10500	10500	10500
8	MK_MAN	13000	13000	13000	13000
9	AD_PRES	24000	24000	24000	24000
10	SA_REP	11000	7000	26600	8867
11	MK_REP	6000	6000	6000	6000
12	ST_CLERK	3500	2500	11700	2925

6. Erstellen Sie eine Abfrage, um die Anzahl der Personen mit derselben Tätigkeit anzuzeigen.

	JOB_ID	COUNT(*)
1	AC_ACCOUNT	1
2	AC_MGR	1
3	AD_ASST	1
4	AD_PRES	1
5	AD_VP	2
6	IT_PROG	3
7	MK_MAN	1
8	MK_REP	1
9	SA_MAN	1
10	SA_REP	3
11	ST_CLERK	4
12	ST_MAN	1

Verallgemeinern Sie die Abfrage, sodass der Benutzer in der Personalabteilung aufgefordert wird, eine Tätigkeitsbezeichnung einzugeben. Speichern Sie das Skript in der Datei lab\_06\_06.sql. Führen Sie die Abfrage aus. Wenn Sie zur Eingabe aufgefordert werden, geben Sie IT\_PROG ein.

	JOB_ID	COUNT(*)
1	IT_PROG	3

7. Bestimmen Sie die Anzahl der Manager, ohne sie aufzulisten. Nennen Sie die Spalte Number of Managers.

**Tipp:** Verwenden Sie die Spalte MANAGER\_ID, um die Anzahl der Manager zu ermitteln.

	Number of Managers
1	8

8. Ermitteln Sie die Differenz zwischen dem höchsten und dem niedrigsten Gehalt. Nennen Sie die Spalte DIFFERENCE.

	DIFFERENCE
1	21500






Führen Sie die folgenden Übungen durch, falls Sie noch Zeit haben:

9. Erstellen Sie einen Bericht, um die Managernummer und das niedrigste Gehalt unter den diesem Manager unterstellten Mitarbeitern anzuzeigen. Schließen Sie alle Mitarbeiter aus, deren Manager nicht bekannt ist. Schließen Sie alle Gruppen aus, deren Mindestgehalt maximal \$ 6.000 beträgt. Sortieren Sie die Ausgabe in absteigender Reihenfolge nach Gehalt.







	MANAGER_ID	MIN(SALARY)
1	102	9000
2	205	8300
3	149	7000

Wenn Sie eine weitere Herausforderung suchen, bearbeiten Sie folgende Aufgaben:

10. Erstellen Sie eine Abfrage, um die Gesamtzahl der Mitarbeiter und daraus jeweils die Anzahl der Mitarbeiter anzuzeigen, die in den Jahren 2005, 2006, 2007 und 2008 eingestellt wurden. Wählen Sie geeignete Spaltenüberschriften.

	 TOTAL	 2005	 2006	 2007	 2008
1	20	3	4	4	1

11. Erstellen Sie eine Matrixabfrage, um für die Abteilungen 20, 50, 80 und 90 die Tätigkeit, das Gehalt für die Tätigkeit nach Abteilungsnummer sowie das Gesamtgehalt für die Tätigkeit anzuzeigen. Wählen Sie geeignete Spaltenüberschriften.

 Job	 Dept 20	 Dept 50	 Dept 80	 Dept 90	 Total
1 IT_PROG	(null)	(null)	(null)	(null)	19200
2 AC_MGR	(null)	(null)	(null)	(null)	12008
3 AC_ACCOUNT	(null)	(null)	(null)	(null)	8300
4 ST_MAN	(null)	5800	(null)	(null)	5800
5 AD_ASST	(null)	(null)	(null)	(null)	4400
6 AD_VP	(null)	(null)	(null)	34000	34000
7 SA_MAN	(null)	(null)	10500	(null)	10500
8 MK_MAN	13000	(null)	(null)	(null)	13000
9 AD_PRES	(null)	(null)	(null)	24000	24000
10 SA_REP	(null)	(null)	19600	(null)	26600
11 MK_REP	6000	(null)	(null)	(null)	6000
12 ST_CLERK	(null)	11700	(null)	(null)	11700

## Übung 1 zu Lektion 6 – Lösung: Mit Gruppenfunktionen Berichte zu aggregierten Daten erstellen

---

Prüfen Sie die Richtigkeit der folgenden Aussagen. Kreuzen Sie Richtig oder Falsch an.

1. Gruppenfunktionen bearbeiten viele Zeilen, um ein Ergebnis pro Gruppe auszugeben.  
**Richtig/Falsch**
2. Gruppenfunktionen berücksichtigen Nullwerte in den Berechnungen.  
**Richtig/Falsch**
3. Die Klausel `WHERE` schränkt die Zeilen vor der Aufnahme in eine Gruppenberechnung ein.  
**Richtig/Falsch**

Die Personalabteilung benötigt folgende Berichte:

4. Es sollen das höchste Gehalt, das niedrigste Gehalt, die Summe der Gehälter und das Durchschnittsgehalt für alle Mitarbeiter ermittelt werden. Nennen Sie die Spalten `Maximum`, `Minimum`, `Sum` und `Average`. Runden Sie die Ergebnisse auf die nächste Ganzzahl. Speichern Sie die SQL-Anweisung als `lab_06_04.sql`. Führen Sie die Abfrage aus.

```
SELECT ROUND(MAX(salary),0) "Maximum",  
       ROUND(MIN(salary),0) "Minimum",  
       ROUND(SUM(salary),0) "Sum",  
       ROUND(AVG(salary),0) "Average"  
FROM   employees;
```

5. Ändern Sie die Abfrage in der Datei `lab_06_04.sql`, um das niedrigste Gehalt, das höchste Gehalt, die Summe der Gehälter und das Durchschnittsgehalt für jede Tätigkeitsbezeichnung anzuzeigen. Speichern Sie die Datei `lab_06_04.sql` unter dem Namen `lab_06_05.sql`. Führen Sie die Anweisung in `lab_06_05.sql` aus.

```
SELECT job_id, ROUND(MAX(salary),0) "Maximum",  
       ROUND(MIN(salary),0) "Minimum",  
       ROUND(SUM(salary),0) "Sum",  
       ROUND(AVG(salary),0) "Average"  
FROM   employees  
GROUP BY job_id;
```

6. Erstellen Sie eine Abfrage, um die Anzahl der Personen mit derselben Tätigkeit anzuzeigen.

```
SELECT job_id, COUNT(*)  
FROM   employees  
GROUP BY job_id;
```

Verallgemeinern Sie die Abfrage, sodass der Benutzer in der Personalabteilung zur Eingabe einer Tätigkeitsbezeichnung aufgefordert wird. Speichern Sie das Skript in der Datei `lab_06_06.sql`. Führen Sie die Abfrage aus. Wenn Sie zur Eingabe aufgefordert werden, geben Sie `IT_PROG` ein, und klicken Sie auf **OK**.

```
SELECT job_id, COUNT(*)  
FROM   employees  
WHERE  job_id = '&job_title'  
GROUP BY job_id;
```

7. Bestimmen Sie die Anzahl der Manager, ohne sie aufzulisten. Nennen Sie die Spalte Number of Managers.

**Tipp:** Verwenden Sie die Spalte `MANAGER_ID`, um die Anzahl der Manager zu ermitteln.

```
SELECT COUNT(DISTINCT manager_id) "Number of Managers"
FROM employees;
```

8. Ermitteln Sie die Differenz zwischen dem höchsten und dem niedrigsten Gehalt. Nennen Sie die Spalte `DIFFERENCE`.

```
SELECT MAX(salary) - MIN(salary) DIFFERENCE
FROM employees;
```

Führen Sie die folgenden Übungen durch, falls Sie noch Zeit haben:

9. Erstellen Sie einen Bericht, um die Managernummer und das niedrigste Gehalt unter den diesem Manager unterstellten Mitarbeitern anzuzeigen. Schließen Sie alle Mitarbeiter aus, deren Manager nicht bekannt ist. Schließen Sie alle Gruppen aus, deren Mindestgehalt maximal \$ 6.000 beträgt. Sortieren Sie die Ausgabe in absteigender Reihenfolge nach Gehalt.

```
SELECT manager_id, MIN(salary)
FROM employees
WHERE manager_id IS NOT NULL
GROUP BY manager_id
HAVING MIN(salary) > 6000
ORDER BY MIN(salary) DESC;
```

Wenn Sie eine weitere Herausforderung suchen, bearbeiten Sie folgende Aufgaben:

10. Erstellen Sie eine Abfrage, um die Gesamtzahl der Mitarbeiter und daraus jeweils die Anzahl der Mitarbeiter anzuzeigen, die in den Jahren 2005, 2006, 2007 und 2008 eingestellt wurden. Wählen Sie geeignete Spaltenüberschriften.

```
SELECT COUNT(*) total,
       SUM(DECODE(TO_CHAR(hire_date, 'YYYY'), 2005, 1, 0)) "2005",
       SUM(DECODE(TO_CHAR(hire_date, 'YYYY'), 2006, 1, 0)) "2006",
       SUM(DECODE(TO_CHAR(hire_date, 'YYYY'), 2007, 1, 0)) "2007",
       SUM(DECODE(TO_CHAR(hire_date, 'YYYY'), 2008, 1, 0)) "2008"
FROM employees;
```

11. Erstellen Sie eine Matrixabfrage, um für die Abteilungen 20, 50, 80 und 90 die Tätigkeit, das Gehalt für die Tätigkeit nach Abteilungsnummer sowie das Gesamtgehalt für die Tätigkeit anzuzeigen. Wählen Sie geeignete Spaltenüberschriften.

```
SELECT  job_id "Job",
        SUM(DECODE(department_id , 20, salary)) "Dept 20",
        SUM(DECODE(department_id , 50, salary)) "Dept 50",
        SUM(DECODE(department_id , 80, salary)) "Dept 80",
        SUM(DECODE(department_id , 90, salary)) "Dept 90",
        SUM(salary) "Total"
FROM    employees
GROUP BY job_id;
```



# **Übungen zu Lektion 7 – Daten aus mehreren Tabellen mit Joins anzeigen**

## **Kapitel 7**

## Übungen zu Lektion 7 – Überblick

---

### Übungsüberblick

Diese Übung behandelt folgende Themen:

- Tabellen mit Equi Joins verknüpfen
- Outer Joins und Self Joins ausführen
- Bedingungen hinzufügen

## Übung 1 zu Lektion 7 – Daten aus mehreren Tabellen mit Joins anzeigen

---

### Überblick

In den Übungen extrahieren Sie Daten aus mehreren Tabellen mithilfe von SQL:1999-konformen Joins.

### Aufgaben

1. Erstellen Sie eine Abfrage für die Personalabteilung, um die Adressen aller Abteilungen auszugeben. Verwenden Sie dazu die Tabellen `LOCATIONS` und `COUNTRIES`. In der Ausgabe sollen Standortkennung (`LOCATION_ID`), Straße, Ort, Bundesstaat/Provinz sowie Land angezeigt werden. Verwenden Sie einen `NATURAL JOIN`, um diese Ausgabe zu erzeugen.

	LOCATION_ID	STREET_ADDRESS	CITY	STATE_PROVINCE	COUNTRY_NAME
1	1400	2014 Jabbawocky Rd	Southlake	Texas	United States of America
2	1500	2011 Interiors Blvd	South San Francisco	California	United States of America
3	1700	2004 Charade Rd	Seattle	Washington	United States of America
4	1800	460 Bloor St. W.	Toronto	Ontario	Canada
5	2500	Magdalen Centre, The Oxford Science Park	Oxford	Oxford	United Kingdom

2. Die Personalabteilung benötigt einen Bericht aller Mitarbeiter mit den zugehörigen Abteilungen. Erstellen Sie eine Abfrage, um Nachname, Abteilungsnummer und Abteilungsname für diese Mitarbeiter anzuzeigen.

	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
1	Abel	80	Sales
2	Davies	50	Shipping
3	De Haan	90	Executive
4	Ernst	60	IT
5	Fay	20	Marketing
6	Gietz	110	Accounting
7	Hartstein	20	Marketing
8	Higgins	110	Accounting
9	Hunold	60	IT
10	King	90	Executive
11	Kochhar	90	Executive
12	Lorentz	60	IT
13	Matos	50	Shipping
14	Mourgos	50	Shipping
15	Rajs	50	Shipping
16	Taylor	80	Sales
17	Vargas	50	Shipping
18	Whalen	10	Administration
19	Zlotkey	80	Sales

3. Die Personalabteilung benötigt einen Bericht zu den Mitarbeitern in Toronto. Zeigen Sie Nachname, Tätigkeits-ID, Abteilungsnummer und Abteilungsname für alle Mitarbeiter an, die in Toronto arbeiten.

	LAST_NAME	JOB_ID	DEPARTMENT_ID	DEPARTMENT_NAME
1	Hartstein	MK_MAN	20	Marketing
2	Fay	MK_REP	20	Marketing

4. Erstellen Sie einen Bericht, um Nachname und Personalnummer jedes Mitarbeiters zusammen mit dem Nachnamen und der Nummer des jeweils zuständigen Managers anzuzeigen. Nennen Sie die Spalten Employee, Emp#, Manager und Mgr#. Speichern Sie die SQL-Anweisung als lab\_07\_04.sql. Führen Sie die Abfrage aus.

	Employee	EMP#	Manager	Mgr#
1	Hunold	103	De Haan	102
2	Fay	202	Hartstein	201
3	Gietz	206	Higgins	205
4	Lorentz	107	Hunold	103
5	Ernst	104	Hunold	103
6	Hartstein	201	King	100
7	Zlotkey	149	King	100
8	Mourgos	124	King	100
9	De Haan	102	King	100
10	Kochhar	101	King	100
11	Higgins	205	Kochhar	101
12	Whalen	200	Kochhar	101
13	Vargas	144	Mourgos	124
14	Matos	143	Mourgos	124
15	Davies	142	Mourgos	124
16	Rajs	141	Mourgos	124
17	Grant	178	Zlotkey	149
18	Taylor	176	Zlotkey	149
19	Abel	174	Zlotkey	149

5. Ändern Sie lab\_07\_04.sql so ab, dass alle Mitarbeiter angezeigt werden, einschließlich King, dem kein Manager zugeordnet ist. Sortieren Sie die Ergebnisse nach der Personalnummer. Speichern Sie die SQL-Anweisung als lab\_07\_05.sql. Führen Sie die Abfrage in lab\_07\_05.sql aus.

	Employee	EMP#	Manager	Mgr#
1	King	100	(null)	(null)
2	Kochhar	101	King	100
3	De Haan	102	King	100
4	Hunold	103	De Haan	102
5	Ernst	104	Hunold	103
6	Lorentz	107	Hunold	103

...

16 Whalen	200 Kochhar	101
17 Hartstein	201 King	100
18 Fay	202 Hartstein	201
19 Higgins	205 Kochhar	101
20 Gietz	206 Higgins	205

6. Erstellen Sie einen Bericht für die Personalabteilung, der die Nachnamen und Abteilungsnummern der Mitarbeiter sowie alle Mitarbeiter anzeigt, die in derselben Abteilung wie ein angegebener Mitarbeiter arbeiten. Geben Sie den Spalten geeignete Namen. Speichern Sie das Skript in der Datei `lab_07_06.sql`.

	A Z	DEPARTMENT	A Z	EMPLOYEE	A Z	COLLEAGUE
1		20	Fay		Hartstein	
2		20	Hartstein		Fay	
3		50	Davies		Matos	
4		50	Davies		Mourgos	
5		50	Davies		Rajs	
6		50	Davies		Vargas	
7		50	Matos		Davies	

...

37		90	King		De Haan	
38		90	King		Kochhar	
39		90	Kochhar		De Haan	
40		90	Kochhar		King	
41		110	Gietz		Higgins	
42		110	Higgins		Gietz	

7. Die Personalabteilung benötigt einen Bericht zu den Gehaltsstufen und Gehältern. Um sich mit der Tabelle `JOB_GRADES` vertraut zu machen, zeigen Sie zunächst ihre Struktur an. Erstellen Sie anschließend eine Abfrage, die Name, Tätigkeits-ID, Abteilungsname, Gehalt und Gehaltsstufe aller Mitarbeiter anzeigt.

```
DESC JOB_GRADES
Name          Null Type
-----
GRADE_LEVEL   VARCHAR2(3)
LOWEST_SAL     NUMBER
HIGHEST_SAL    NUMBER
```

	A Z	LAST_NAME	A Z	JOB_ID	A Z	DEPARTMENT_NAME	A Z	SALARY	A Z	GRADE_LEVEL
1		King		AD_PRES		Executive		24000		E
2		Kochhar		AD_VP		Executive		17000		E
3		De Haan		AD_VP		Executive		17000		E
4		Hartstein		MK_MAN		Marketing		13000		D
5		Higgins		AC_MGR		Accounting		12008		D
6		Abel		SA_REP		Sales		11000		D
7		Zlotkey		SA_MAN		Sales		10500		D
8		Hunold		IT_PROG		IT		9000		C
9		Taylor		SA_REP		Sales		8600		C
10		Gietz		AC_ACCOUNT		Accounting		8300		C
11		Ernst		IT_PROG		IT		6000		C
12		Fay		MK_REP		Marketing		6000		C
13		Mourgos		ST_MAN		Shipping		5800		B
14		Whalen		AD_ASST		Administration		4400		B
15		Lorentz		IT_PROG		IT		4200		B
16		Rajs		ST_CLERK		Shipping		3500		B
17		Davies		ST_CLERK		Shipping		3100		B
18		Matos		ST_CLERK		Shipping		2600		A
19		Vargas		ST_CLERK		Shipping		2500		A

Wenn Sie eine weitere Herausforderung suchen, bearbeiten Sie folgende Aufgaben:

- Die Personalabteilung möchte die Namen aller Mitarbeiter ermitteln, die nach Davies eingestellt wurden. Erstellen Sie eine Abfrage, um Name und Einstellungsdatum aller Mitarbeiter anzuzeigen, die nach dem Mitarbeiter Davies eingestellt wurden.

	A Z	LAST_NAME	A Z	HIRE_DATE
1		Kochhar		21-SEP-05
2		Hunold		03-JAN-06
3		Ernst		21-MAY-07
4		Lorentz		07-FEB-07
5		Mourgos		16-NOV-07
6		Matos		15-MAR-06
7		Vargas		09-JUL-06
8		Zlotkey		29-JAN-08
9		Taylor		24-MAR-06
10		Grant		24-MAY-07
11		Fay		17-AUG-05

9. Die Personalabteilung benötigt Name und Einstellungsdatum aller Mitarbeiter, die vor ihren Managern eingestellt wurden, sowie Name und Einstellungsdatum der zugehörigen Manager. Speichern Sie das Skript in der Datei lab\_07\_09.sql.

	LAST_NAME	HIRE_DATE	LAST_NAME_1	HIRE_DATE_1
1	De Haan	13-JAN-01	King	17-JUN-03
2	Higgins	07-JUN-02	Kochhar	21-SEP-05
3	Whalen	17-SEP-03	Kochhar	21-SEP-05
4	Vargas	09-JUL-06	Mourgos	16-NOV-07
5	Matos	15-MAR-06	Mourgos	16-NOV-07
6	Davies	29-JAN-05	Mourgos	16-NOV-07
7	Rajs	17-OCT-03	Mourgos	16-NOV-07
8	Grant	24-MAY-07	Zlotkey	29-JAN-08
9	Taylor	24-MAR-06	Zlotkey	29-JAN-08
10	Abel	11-MAY-04	Zlotkey	29-JAN-08

## Übung 1 zu Lektion 7 – Lösung: Daten aus mehreren Tabellen mit Joins anzeigen

---

1. Erstellen Sie eine Abfrage für die Personalabteilung, um die Adressen aller Abteilungen auszugeben. Verwenden Sie dazu die Tabellen `LOCATIONS` und `COUNTRIES`. In der Ausgabe sollen Standortkennung, Straße, Ort, Bundesstaat/Provinz sowie Land angezeigt werden. Verwenden Sie einen `NATURAL JOIN`, um diese Ausgabe zu erzeugen.

```
SELECT location_id, street_address, city, state_province,
country_name
FROM   locations
NATURAL JOIN countries;
```

2. Die Personalabteilung benötigt einen Bericht aller Mitarbeiter mit den zugehörigen Abteilungen. Erstellen Sie eine Abfrage, um Nachname, Abteilungsnummer und Abteilungsname für diese Mitarbeiter anzuzeigen.

```
SELECT last_name, department_id, department_name
FROM   employees
JOIN   departments
USING (department_id);
```

3. Die Personalabteilung benötigt einen Bericht zu den Mitarbeitern in Toronto. Zeigen Sie Nachname, Tätigkeits-ID, Abteilungsnummer und Abteilungsname für alle Mitarbeiter an, die in Toronto arbeiten.

```
SELECT e.last_name, e.job_id, e.department_id, d.department_name
FROM   employees e JOIN departments d
ON     (e.department_id = d.department_id)
JOIN   locations l
USING (location_id)
WHERE  LOWER(l.city) = 'toronto';
```

4. Erstellen Sie einen Bericht, um Nachname und Personalnummer jedes Mitarbeiters zusammen mit dem Nachnamen und der Nummer des jeweils zuständigen Managers anzuzeigen. Nennen Sie die Spalten `Employee`, `Emp#`, `Manager` und `Mgr#`. Speichern Sie die SQL-Anweisung als `lab_07_04.sql`. Führen Sie die Abfrage aus.

```
SELECT w.last_name "Employee", w.employee_id "EMP#",
       m.last_name "Manager", m.employee_id  "Mgr#"
FROM   employees w JOIN employees m
ON     (w.manager_id = m.employee_id);
```

5. Ändern Sie `lab_07_04.sql` so ab, dass alle Mitarbeiter angezeigt werden, einschließlich King, dem kein Manager zugeordnet ist. Sortieren Sie die Ergebnisse nach der Personalnummer. Speichern Sie die SQL-Anweisung als `lab_07_05.sql`. Führen Sie die Abfrage in `lab_07_05.sql` aus.

```
SELECT w.last_name "Employee", w.employee_id "EMP#",
       m.last_name "Manager", m.employee_id  "Mgr#"
FROM   employees w
```



```

LEFT    OUTER JOIN employees m
ON      (w.manager_id = m.employee_id)
ORDER BY 2;

```

6. Erstellen Sie einen Bericht für die Personalabteilung, der die Nachnamen und Abteilungsnummern der Mitarbeiter sowie alle Mitarbeiter anzeigt, die in derselben Abteilung wie ein angegebener Mitarbeiter arbeiten. Geben Sie den Spalten geeignete Namen. Speichern Sie das Skript in der Datei lab\_07\_06.sql. Führen Sie die Abfrage aus.

```

SELECT e.department_id department, e.last_name employee,
       c.last_name colleague
FROM   employees e JOIN employees c
ON     (e.department_id = c.department_id)
WHERE  e.employee_id <> c.employee_id
ORDER BY e.department_id, e.last_name, c.last_name;

```

7. Die Personalabteilung benötigt einen Bericht zu den Gehaltsstufen und Gehältern. Um sich mit der Tabelle JOB\_GRADES vertraut zu machen, zeigen Sie zunächst ihre Struktur an. Erstellen Sie anschließend eine Abfrage, die Name, Tätigkeits-ID, Abteilungsname, Gehalt und Gehaltsstufe aller Mitarbeiter anzeigt.

```

DESC JOB_GRADES
/
SELECT e.last_name, e.job_id, d.department_name,
       e.salary, j.grade_level
FROM   employees e JOIN departments d
ON     (e.department_id = d.department_id)
JOIN   job_grades j
ON     (e.salary BETWEEN j.lowest_sal AND j.highest_sal);

```

Wenn Sie eine weitere Herausforderung suchen, bearbeiten Sie folgende Aufgaben:

8. Die Personalabteilung möchte die Namen aller Mitarbeiter ermitteln, die nach Davies eingestellt wurden. Erstellen Sie eine Abfrage, um Name und Einstellungsdatum aller Mitarbeiter anzuzeigen, die nach dem Mitarbeiter Davies eingestellt wurden.

```

SELECT e.last_name, e.hire_date
FROM   employees e JOIN employees davies
ON     (davies.last_name = 'Davies')
WHERE  davies.hire_date < e.hire_date;

```

9. Die Personalabteilung benötigt Name und Einstellungsdatum aller Mitarbeiter, die vor ihren Managern eingestellt wurden, sowie Name und Einstellungsdatum der zugehörigen Manager. Speichern Sie das Skript in der Datei lab\_07\_09.sql.

```

SELECT w.last_name, w.hire_date, m.last_name, m.hire_date
FROM   employees w JOIN employees m
ON     (w.manager_id = m.employee_id)
WHERE  w.hire_date < m.hire_date;

```



# **Übungen zu Lektion 8 – Unterabfragen in Abfragen**

## **Kapitel 8**

## Übungen zu Lektion 8 – Überblick

---

### Übungsüberblick

Diese Übung behandelt folgende Themen:

- Unterabfragen erstellen, um Werte basierend auf unbekannten Kriterien abzufragen
- Mithilfe von Unterabfragen Werte ermitteln, die in einer Datengruppe, nicht aber in einer anderen Datengruppe vorkommen

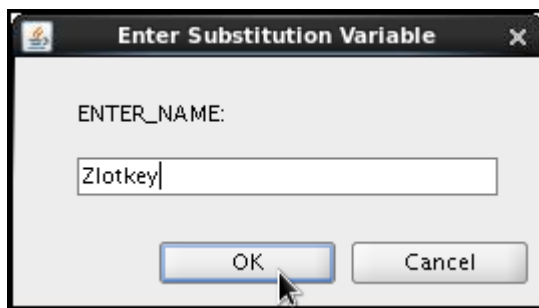
# Übung 1 zu Lektion 8 – Unterabfragen in Abfragen

## Überblick

In diesen Übungen erstellen Sie komplexe Abfragen mit verschachtelten `SELECT`-Anweisungen. Aus praktischen Gründen erstellen Sie zuerst die innere Abfrage. Prüfen Sie, ob die Abfrage korrekt ausgeführt wird und die erwarteten Ergebnisse liefert, bevor Sie die äußere Abfrage erstellen.

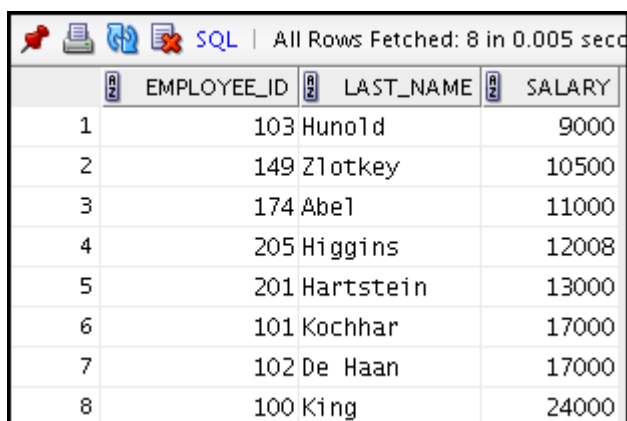
## Aufgaben

1. Die Personalabteilung benötigt eine Abfrage, die den Benutzer auffordert, den Nachnamen eines Mitarbeiters einzugeben. Die Abfrage soll dann Nachname und Einstellungsdatum für jeden Mitarbeiter anzeigen, der in derselben Abteilung arbeitet wie der vom Benutzer angegebene Mitarbeiter (mit Ausnahme dieses Mitarbeiters). Beispiel: Wenn der Benutzer Zlotkey eingibt, sollen alle Mitarbeiter ermittelt werden, die mit Zlotkey arbeiten (mit Ausnahme von Zlotkey).



	LAST_NAME	HIRE_DATE
1	Abel	11-MAY-04
2	Taylor	24-MAR-06

2. Erstellen Sie einen Bericht, um Personalnummer, Nachname und Gehalt aller Mitarbeiter anzuzeigen, die mehr als das Durchschnittsgehalt verdienen. Sortieren Sie die Ergebnisse in aufsteigender Reihenfolge nach Gehalt.



	EMPLOYEE_ID	LAST_NAME	SALARY
1	103	Hunold	9000
2	149	Zlotkey	10500
3	174	Abel	11000
4	205	Higgins	12008
5	201	Hartstein	13000
6	101	Kochhar	17000
7	102	De Haan	17000
8	100	King	24000

- Erstellen Sie eine Abfrage, die Personalnummer und Nachname aller Mitarbeiter anzeigt, die in einer Abteilung mit einem Mitarbeiter arbeiten, dessen Nachname ein "u" enthält. Speichern Sie die SQL-Anweisung als `lab_08_03.sql`. Führen Sie die Abfrage aus.

	EMPLOYEE_ID	LAST_NAME
1	124	Mourgos
2	141	Rajs
3	142	Davies
4	143	Matos
5	144	Vargas
6	103	Hunold
7	104	Ernst
8	107	Lorentz

- Die Personalabteilung benötigt einen Bericht, der Nachname, Abteilungsnummer und Tätigkeits-ID aller Mitarbeiter anzeigt, die in der Abteilung mit der Standortkennung 1700 arbeiten.

	LAST_NAME	DEPARTMENT_ID	JOB_ID
1	Whalen	10	AD_ASST
2	King	90	AD_PRES
3	Kochhar	90	AD_VP
4	De Haan	90	AD_VP
5	Higgins	110	AC_MGR
6	Gietz	110	AC_ACCOUNT

Ändern Sie die Abfrage so, dass der Benutzer zur Eingabe einer Standortkennung aufgefordert wird. Speichern Sie die Abfrage in einer Datei mit dem Namen `lab_08_04.sql`.

- Erstellen Sie für die Personalabteilung einen Bericht, der Nachname und Gehalt für jeden Mitarbeiter anzeigt, der King unterstellt ist.

	LAST_NAME	SALARY
1	Kochhar	17000
2	De Haan	17000
3	Mourgos	5800
4	Zlotkey	10500
5	Hartstein	13000

- Erstellen Sie für die Personalabteilung einen Bericht, der für jeden Mitarbeiter aus der Abteilung "Executive" Abteilungsnummer, Nachname und Tätigkeits-ID anzeigt.

	DEPARTMENT_ID	LAST_NAME	JOB_ID
1	90	King	AD_PRES
2	90	Kochhar	AD_VP
3	90	De Haan	AD_VP

7. Erstellen Sie einen Bericht, der eine Liste aller Mitarbeiter anzeigt, deren Gehalt über dem Gehalt der Mitarbeiter aus Abteilung 60 liegt.

	LAST_NAME
1	King
2	Kochhar
3	De Haan
4	Hartstein
5	Hunold
6	Higgins
7	Abel
8	Zlotkey
9	Taylor
10	Gietz
11	Grant
12	Fay
13	Ernst
14	Mourgos
15	Whalen

Führen Sie die folgende Übung durch, falls Sie noch Zeit haben:

8. Ändern Sie die Abfrage in `lab_08_03.sql` so ab, dass Personalnummer, Nachname und Gehalt aller Mitarbeiter angezeigt werden, die mehr als das Durchschnittsgehalt verdienen und in derselben Abteilung wie ein Mitarbeiter arbeiten, dessen Nachname ein "u" enthält. Speichern Sie die Datei `lab_08_03.sql` als `lab_08_08.sql`. Führen Sie die Anweisung in `lab_08_08.sql` aus.

	EMPLOYEE_ID	LAST_NAME	SALARY
1	103	Hunold	9000

## Übung 1 zu Lektion 8 – Lösung: Unterabfragen in Abfragen

---

1. Die Personalabteilung benötigt eine Abfrage, die den Benutzer zur Eingabe des Nachnamens eines Mitarbeiters auffordert. Die Abfrage soll dann Nachname und Einstellungsdatum für jeden Mitarbeiter anzeigen, der in derselben Abteilung arbeitet wie der vom Benutzer angegebene Mitarbeiter (mit Ausnahme dieses Mitarbeiters). Beispiel: Wenn der Benutzer Zlotkey eingibt, sollen alle Mitarbeiter ermittelt werden, die mit Zlotkey arbeiten (mit Ausnahme von Zlotkey).

```
--Execute the UNDEFINE command to remove a variable

UNDEFINE Enter_name

-- Execute the below SELECT statements to retrieve the values
from employees table

SELECT last_name, hire_date
FROM   employees
WHERE  department_id = (SELECT department_id
                        FROM   employees
                        WHERE  last_name = '&Enter_name')
AND    last_name <> '&Enter_name';
```

**Hinweis:** Bei UNDEFINE und SELECT handelt es sich um einzelne Abfragen. Führen Sie sie nacheinander aus, oder drücken Sie Strg+A+F9, um sie gleichzeitig auszuführen.

2. Erstellen Sie einen Bericht, um Personalnummer, Nachname und Gehalt aller Mitarbeiter anzuzeigen, die mehr als das Durchschnittsgehalt verdienen. Sortieren Sie die Ergebnisse in aufsteigender Reihenfolge nach Gehalt.

```
SELECT employee_id, last_name, salary
FROM   employees
WHERE  salary > (SELECT AVG(salary)
                FROM   employees)
ORDER BY salary;
```

3. Erstellen Sie eine Abfrage, die Personalnummer und Nachname aller Mitarbeiter anzeigt, die in einer Abteilung mit einem Mitarbeiter arbeiten, dessen Nachname ein "u" enthält. Speichern Sie die SQL-Anweisung als lab\_08\_03.sql. Führen Sie die Abfrage aus.

```
SELECT employee_id, last_name
FROM   employees
WHERE  department_id IN (SELECT department_id
                        FROM   employees
                        WHERE  last_name like '%u%');
```



4. Die Personalabteilung benötigt einen Bericht, der Nachname, Abteilungsnummer und Tätigkeits-ID aller Mitarbeiter anzeigt, die in der Abteilung mit der Standortkennung 1700 arbeiten.

```
SELECT last_name, department_id, job_id
FROM employees
WHERE department_id IN (SELECT department_id
                        FROM departments
                        WHERE location_id = 1700);
```

Ändern Sie die Abfrage so, dass der Benutzer zur Eingabe einer Standortkennung aufgefordert wird. Speichern Sie die Abfrage in einer Datei mit dem Namen lab\_08\_04.sql.

```
SELECT last_name, department_id, job_id
FROM employees
WHERE department_id IN (SELECT department_id
                        FROM departments
                        WHERE location_id =
                        &Enter_location);
```

5. Erstellen Sie für die Personalabteilung einen Bericht, der Nachname und Gehalt für jeden Mitarbeiter anzeigt, der King unterstellt ist.

```
SELECT last_name, salary
FROM employees
WHERE manager_id = (SELECT employee_id
                    FROM employees
                    WHERE last_name = 'King');
```

6. Erstellen Sie für die Personalabteilung einen Bericht, der für jeden Mitarbeiter aus der Abteilung "Executive" Abteilungsnummer, Nachname und Tätigkeits-ID anzeigt.

```
SELECT department_id, last_name, job_id
FROM employees
WHERE department_id IN (SELECT department_id
                        FROM departments
                        WHERE department_name =
                        'Executive');
```

7. Erstellen Sie einen Bericht, der eine Liste aller Mitarbeiter anzeigt, deren Gehalt über dem Gehalt der Mitarbeiter aus Abteilung 60 liegt.

```
SELECT last_name FROM employees
WHERE salary > ANY (SELECT salary
                    FROM employees
                    WHERE department_id=60);
```

Führen Sie die folgende Übung durch, falls Sie noch Zeit haben:

8. Ändern Sie die Abfrage in `lab_08_03.sql` so ab, dass Personalnummer, Nachname und Gehalt aller Mitarbeiter angezeigt werden, die mehr als das Durchschnittsgehalt verdienen und in derselben Abteilung wie ein Mitarbeiter arbeiten, dessen Nachname ein "u" enthält. Speichern Sie die Datei `lab_08_03.sql` als `lab_08_08.sql`. Führen Sie die Anweisung in `lab_08_08.sql` aus.

```
SELECT employee_id, last_name, salary
FROM   employees
WHERE  department_id IN (SELECT department_id
                        FROM   employees
                        WHERE  last_name like '%u%')
AND    salary > (SELECT AVG(salary)
                FROM   employees);
```

# **Übungen zu Lektion 9 – Mengenoperatoren**

## **Kapitel 9**

## Übungen zu Lektion 9 – Überblick

---

### Übungsüberblick

In dieser Übung erstellen Sie Berichte mit den folgenden Operatoren:

- UNION
- INTERSECT
- MINUS

## Übung 1 zu Lektion 9 – Mengenoperatoren

### Überblick

In dieser Übung erstellen Sie Abfragen mit den Mengenoperatoren UNION, INTERSECT und MINUS.

### Aufgaben

1. Die Personalabteilung benötigt eine Liste der Abteilungsnummern für Abteilungen, die nicht die Tätigkeits-ID ST\_CLERK enthalten. Erstellen Sie diesen Bericht mithilfe der Mengenoperatoren.

	DEPARTMENT_ID
1	10
2	20
3	60
4	80
5	90
6	110
7	190

2. Die Personalabteilung benötigt eine Liste der Länder, in denen es keine Abteilungen gibt. Zeigen Sie die Kennung und den Namen der Länder an. Erstellen Sie diesen Bericht mithilfe der Mengenoperatoren.

	COUNTRY_ID	COUNTRY_NAME
1	DE	Germany

3. Erstellen Sie eine Liste aller Mitarbeiter, die in den Abteilungen 50 und 80 arbeiten. Zeigen Sie mithilfe der Mengenoperatoren Personalnummer, Tätigkeits-ID und Abteilungsnummer an.

	EMPLOYEE_ID	JOB_ID	DEPARTMENT_ID
1	124	ST_MAN	50
2	141	ST_CLERK	50
3	142	ST_CLERK	50
4	143	ST_CLERK	50
5	144	ST_CLERK	50
6	149	SA_MAN	80
7	174	SA_REP	80
8	176	SA_REP	80

4. Erstellen Sie einen Bericht, der die Details aller Mitarbeiter mit der Tätigkeit Sales Representative auflistet, die aktuell in der Vertriebsabteilung arbeiten.

	EMPLOYEE_ID
1	174
2	176

5. Die Personalabteilung benötigt einen Bericht mit den folgenden Angaben:

- Nachname und Abteilungsnummer aller Mitarbeiter aus der Tabelle `EMPLOYEES`, unabhängig davon, ob sie zu einer Abteilung gehören
- Abteilungsnummer und -name aller Abteilungen aus der Tabelle `DEPARTMENTS`, unabhängig davon, ob Mitarbeiter darin arbeiten

Erstellen Sie für diesen Zweck eine zusammengesetzte Abfrage.

	LAST_NAME	DEPARTMENT_ID	DEPT_NAME
1	Abel	80 (null)	
2	Davies	50 (null)	
3	De Haan	90 (null)	
4	Ernst	60 (null)	
5	Fay	20 (null)	
6	Gietz	110 (null)	
7	Grant	(null) (null)	
8	Hartstein	20 (null)	
9	Higgins	110 (null)	
10	Hunold	60 (null)	
11	King	90 (null)	
12	Kochhar	90 (null)	
13	Lorentz	60 (null)	
14	Matos	50 (null)	
15	Mourgos	50 (null)	
16	Rajs	50 (null)	
17	Taylor	80 (null)	
18	Vargas	50 (null)	
19	Whalen	10 (null)	
20	Zlotkey	80 (null)	
21	(null)	10	Administration
22	(null)	20	Marketing
23	(null)	50	Shipping
24	(null)	60	IT
25	(null)	80	Sales
26	(null)	90	Executive
27	(null)	110	Accounting
28	(null)	190	Contracting

## Übung 1 zu Lektion 9 – Lösung: Mengenoperatoren

---

1. Die Personalabteilung benötigt eine Liste der Abteilungsnummern für Abteilungen, die nicht die Tätigkeits-ID `ST_CLERK` enthalten. Erstellen Sie diesen Bericht mithilfe der Mengenoperatoren.

```
SELECT department_id
FROM departments
MINUS
SELECT department_id
FROM employees
WHERE job_id = 'ST_CLERK';
```

2. Die Personalabteilung benötigt eine Liste der Länder, in denen es keine Abteilungen gibt. Zeigen Sie die Kennung und den Namen der Länder an. Erstellen Sie diesen Bericht mithilfe der Mengenoperatoren.

```
SELECT country_id, country_name
FROM countries
MINUS
SELECT l.country_id, c.country_name
FROM locations l JOIN countries c
ON (l.country_id = c.country_id)
JOIN departments d
ON d.location_id=l.location_id;
```

3. Erstellen Sie eine Liste aller Mitarbeiter, die in den Abteilungen 50 und 80 arbeiten. Zeigen Sie mithilfe der Mengenoperatoren Personalnummer, Tätigkeits-ID und Abteilungsnummer an.

```
SELECT employee_id, job_id, department_id
FROM EMPLOYEES
WHERE department_id=50
UNION ALL
SELECT employee_id, job_id, department_id
FROM EMPLOYEES
WHERE department_id=80;
```

4. Erstellen Sie einen Bericht, der die Details aller Mitarbeiter mit der Tätigkeit Sales Representative auflistet, die aktuell in der Vertriebsabteilung arbeiten.

```
SELECT EMPLOYEE_ID
FROM EMPLOYEES
WHERE JOB_ID='SA_REP'
INTERSECT
SELECT EMPLOYEE_ID
FROM EMPLOYEES
WHERE DEPARTMENT_ID=80;
```

5. Die Personalabteilung benötigt einen Bericht mit den folgenden Angaben:

- Nachname und Abteilungsnummer aller Mitarbeiter aus der Tabelle `EMPLOYEES`, unabhängig davon, ob sie zu einer Abteilung gehören
- Abteilungsnummer und -name aller Abteilungen aus der Tabelle `DEPARTMENTS`, unabhängig davon, ob Mitarbeiter darin arbeiten

Erstellen Sie für diesen Zweck eine zusammengesetzte Abfrage.

```
SELECT last_name, department_id, TO_CHAR(null) dept_name
FROM   employees
UNION
SELECT TO_CHAR(null), department_id, department_name
FROM   departments;
```



# **Übungen zu Lektion 10 – Daten bearbeiten**

## **Kapitel 10**

# Übungen zu Lektion 10 – Überblick

---

## Lektionsüberblick

Diese Übung behandelt folgende Themen:

- Zeilen in Tabellen einfügen
- Zeilen einer Tabelle aktualisieren und löschen
- Transaktionen steuern

**Hinweis:** Führen Sie das folgende Skript aus, bevor Sie diese Übung beginnen:

```
/home/oracle/labs/sql1/code_ex /cleanup_scripts/cleanup_10.sql script
```

## Übung 1 zu Lektion 10 – Tabellen mit DML-Anweisungen verwalten

---

### Überblick

Die Personalabteilung möchte, dass Sie SQL-Anweisungen erstellen, mit denen Mitarbeiterdaten eingefügt, aktualisiert und gelöscht werden können. Bevor Sie die Anweisungen an die Personalabteilung übergeben, verwenden Sie die Tabelle `MY_EMPLOYEE` als Prototyp.

### Hinweise

- Um die Abfrage auszuführen, verwenden Sie für alle DML-Anweisungen das Symbol **Run Script** (oder drücken F5). Auf diese Weise werden die Rückmeldungen in der Registerkarte **Script Output** angezeigt. Für `SELECT`-Abfragen verwenden Sie weiterhin das Symbol **Execute Statement** oder drücken F9, um die formatierte Ausgabe in der Registerkarte **Results** anzuzeigen.
- Führen Sie das Skript `cleanup_10.sql` unter `/home/oracle/labs/sql1/code_ex/cleanup_scripts/` aus, bevor Sie mit der Durchführung der folgenden Aufgaben beginnen.

### Aufgaben

1. Erstellen Sie die Tabelle `MY_EMPLOYEE`.
2. Beschreiben Sie die Struktur der Tabelle `MY_EMPLOYEE`, um die Spaltennamen zu identifizieren.

```
DESCRIBE my_employee
Name      Null      Type
-----
ID         NOT NULL   NUMBER(4)
LAST_NAME                VARCHAR2(25)
FIRST_NAME               VARCHAR2(25)
USERID                  VARCHAR2(8)
SALARY                  NUMBER(9,2)
```

- Erstellen Sie eine `INSERT`-Anweisung, um der Tabelle `MY_EMPLOYEE` die *erste Zeile* der folgenden Beispieldaten hinzuzufügen. Listen Sie die Spalten nicht in der Klausel `INSERT` auf. *Geben Sie noch nicht alle Zeilen ein.*

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	Patel	Ralph	rpatel	895
2	Dancs	Betty	bdancs	860
3	Biri	Ben	bbiri	1100
4	Newman	Chad	cnewman	750
5	Ropeburn	Audrey	aropebur	1550

- Füllen Sie die Tabelle `MY_EMPLOYEE` mit der zweiten Zeile der Beispieldaten aus der vorherigen Liste auf. Listen Sie dieses Mal die Spalten explizit in der Klausel `INSERT` auf.
- Prüfen Sie die der Tabelle hinzugefügten Daten.

	ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	1 Patel	Ralph	rpatel		895
2	2 Dancs	Betty	bdancs		860

- Erstellen Sie eine `INSERT`-Anweisung in einer dynamisch wiederverwendbaren Skriptdatei, um die verbleibenden Zeilen in die Tabelle `MY_EMPLOYEE` zu laden. Das Skript soll zur Eingabe aller Spalten (`ID`, `LAST_NAME`, `FIRST_NAME`, `USERID` und `SALARY`) auffordern. Speichern Sie das Skript in der Datei `lab_10_06.sql`.
- Füllen Sie die Tabelle mit den nächsten beiden Zeilen der im 3. Schritt aufgelisteten Beispieldaten auf, indem Sie die Anweisung `INSERT` im erstellten Skript ausführen.
- Prüfen Sie die der Tabelle hinzugefügten Daten.

	ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	1 Patel	Ralph	rpatel		895
2	2 Dancs	Betty	bdancs		860
3	3 Biri	Ben	bbiri		1100
4	4 Newman	Chad	cnewman		750

- Schreiben Sie die hinzugefügten Daten dauerhaft fest.

Aktualisieren und löschen Sie Daten in der Tabelle `MY_EMPLOYEE`.

- Ändern Sie den Nachnamen des dritten Mitarbeiters in Drexler.
- Ändern Sie das Gehalt aller Mitarbeiter, deren Gehalt unter \$ 900 liegt, in \$ 1000.

12. Prüfen Sie die Änderungen an der Tabelle.

	ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	1	Patel	Ralph	rpatel	1000
2	2	Dancs	Betty	bdancs	1000
3	3	Drexler	Ben	bbiri	1100
4	4	Newman	Chad	cnewman	1000

13. Löschen Sie Betty Dancs aus der Tabelle MY\_EMPLOYEE.

14. Prüfen Sie die Änderungen an der Tabelle.

	ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	1	Patel	Ralph	rpatel	1000
2	3	Drexler	Ben	bbiri	1100
3	4	Newman	Chad	cnewman	1000

15. Schreiben Sie alle noch nicht gespeicherten Änderungen fest.

Steuern Sie die Datentransaktion in der Tabelle MY\_EMPLOYEE.

16. Füllen Sie die Tabelle mit der letzten Zeile der aufgelisteten Beispieldaten (3. Schritt) auf. Verwenden Sie hierfür die Anweisungen aus dem Skript, das Sie im 6. Schritt erstellt haben. Führen Sie die Anweisungen im Skript aus.

**Hinweis:** Führen Sie die Schritte (17 - 23) in einer Session durch.

17. Prüfen Sie die der Tabelle hinzugefügten Daten.

	ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	1	Patel	Ralph	rpatel	1000
2	3	Drexler	Ben	bbiri	1100
3	4	Newman	Chad	cnewman	1000
4	5	Ropeburn	Audrey	aropebur	1550

18. Markieren Sie einen Zwischenpunkt in der Verarbeitung der Transaktion.

19. Löschen Sie alle Zeilen aus der Tabelle MY\_EMPLOYEE.

20. Prüfen Sie, ob die Tabelle leer ist.

21. Verwerfen Sie den letzten DELETE-Vorgang, nicht jedoch den vorhergehenden INSERT-Vorgang.

22. Prüfen Sie, ob die neue Zeile weiterhin intakt ist.

	ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	1	Patel	Ralph	rpatel	1000
2	3	Drexler	Ben	bbiri	1100
3	4	Newman	Chad	cnewman	1000
4	5	Ropeburn	Audrey	aropebur	1550

23. Schreiben Sie die hinzugefügten Daten dauerhaft fest.

Führen Sie die folgende Übung durch, falls Sie noch Zeit haben:

24. Ändern Sie das Skript `lab_10_06.sql` so, dass die `USERID` automatisch generiert wird, indem der erste Buchstabe des Vornamens und die ersten sieben Zeichen des Nachnamens verkettet werden. Für die generierte `USERID` muss Kleinschreibung verwendet werden. Daher sollte das Skript nicht zur Eingabe der `USERID` auffordern. Speichern Sie dieses Skript als `lab_10_24.sql`.

25. Führen Sie das Skript `lab_10_24.sql` aus, um den folgenden Datensatz einzufügen:

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
6	Anthony	Mark	manthony	1230

26. Prüfen Sie, ob die neue Zeile mit korrekter `USERID` hinzugefügt wurde.

	ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	6	Anthony	Mark	manthony	1230

## Übung 1 zu Lektion 10 – Lösung: Tabellen mit DML-Anweisungen verwalten

Fügen Sie Daten in die Tabelle MY\_EMPLOYEE ein.

1. Erstellen Sie die Tabelle MY\_EMPLOYEE.

```
CREATE TABLE my_employee  
(id NUMBER(4) CONSTRAINT my_employee_id_pk PRIMARY Key,  
last_name VARCHAR2(25),  
first_name VARCHAR2(25),  
userid VARCHAR2(8),  
salary NUMBER(9,2));
```

2. Beschreiben Sie die Struktur der Tabelle MY\_EMPLOYEE, um die Spaltennamen zu identifizieren.

```
DESCRIBE my_employee
```

3. Erstellen Sie eine INSERT-Anweisung, um der Tabelle MY\_EMPLOYEE die erste Zeile der folgenden Beispieldaten hinzuzufügen. Listen Sie die Spalten nicht in der Klausel INSERT auf.

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
1	Patel	Ralph	rpatel	895
2	Dancs	Betty	bdancs	860
3	Biri	Ben	bbiri	1100
4	Newman	Chad	cnewman	750
5	Ropeburn	Audrey	aropebur	1550

```
INSERT INTO my_employee  
VALUES (1, 'Patel', 'Ralph', 'rpatel', 895);
```

4. Füllen Sie die Tabelle MY\_EMPLOYEE mit der zweiten Zeile der Beispieldaten aus der vorherigen Liste auf. Listen Sie dieses Mal die Spalten explizit in der Klausel INSERT auf.

```
INSERT INTO my_employee (id, last_name, first_name,
                        userid, salary)
VALUES (2, 'Dancs', 'Betty', 'bdancs', 860);
```

5. Prüfen Sie die der Tabelle hinzugefügten Daten.

```
SELECT *
FROM my_employee;
```

6. Erstellen Sie eine INSERT-Anweisung in einer dynamisch wiederverwendbaren Skriptdatei, um die verbleibenden Zeilen in die Tabelle MY\_EMPLOYEE zu laden. Das Skript soll zur Eingabe aller Spalten (ID, LAST\_NAME, FIRST\_NAME, USERID und SALARY) auffordern. Speichern Sie dieses Skript als lab\_10\_06.sql.

```
INSERT INTO my_employee
VALUES (&p_id, '&p_last_name', '&p_first_name',
      '&p_userid', &p_salary);
```

7. Füllen Sie die Tabelle mit den nächsten beiden Zeilen der im 3. Schritt aufgelisteten Beispieldaten auf, indem Sie die Anweisung INSERT im erstellten Skript ausführen.

```
INSERT INTO my_employee
VALUES (&p_id, '&p_last_name', '&p_first_name',
      '&p_userid', &p_salary);
```

8. Prüfen Sie die der Tabelle hinzugefügten Daten.

```
SELECT *
FROM my_employee;
```

9. Schreiben Sie die hinzugefügten Daten dauerhaft fest.

```
COMMIT;
```

Aktualisieren und löschen Sie Daten in der Tabelle MY\_EMPLOYEE.

10. Ändern Sie den Nachnamen des dritten Mitarbeiters in Drexler.

```
UPDATE my_employee
SET last_name = 'Drexler'
WHERE id = 3;
```

11. Ändern Sie das Gehalt aller Mitarbeiter, deren Gehalt unter \$ 900 liegt, in \$ 1000.

```
UPDATE my_employee
SET salary = 1000
WHERE salary < 900;
```



12. Prüfen Sie die Änderungen an der Tabelle.

```
SELECT  *  
FROM    my_employee;
```

13. Löschen Sie Betty Dancs aus der Tabelle MY\_EMPLOYEE.

```
DELETE  
FROM    my_employee  
WHERE last_name = 'Dancs';
```

14. Prüfen Sie die Änderungen an der Tabelle.

```
SELECT  *  
FROM    my_employee;
```

15. Schreiben Sie alle noch nicht gespeicherten Änderungen fest.

```
COMMIT;
```

Steuern Sie die Datentransaktion in der Tabelle MY\_EMPLOYEE.

16. Füllen Sie die Tabelle mit der letzten Zeile der aufgelisteten Beispieldaten (3. Schritt) auf. Verwenden Sie hierfür die Anweisungen aus dem Skript, das Sie im 6. Schritt erstellt haben. Führen Sie die Anweisungen im Skript aus.

```
INSERT INTO my_employee  
VALUES (&p_id, '&p_last_name', '&p_first_name',  
        '&p_userid', &p_salary);
```

**Hinweis:** Führen Sie die Schritte (17 - 23) in einer Session durch.

17. Prüfen Sie die der Tabelle hinzugefügten Daten.

```
SELECT  *  
FROM    my_employee;
```

18. Markieren Sie einen Zwischenpunkt in der Verarbeitung der Transaktion.

```
SAVEPOINT step_17;
```

19. Löschen Sie alle Zeilen aus der Tabelle MY\_EMPLOYEE.

```
DELETE  
FROM    my_employee;
```

20. Prüfen Sie, ob die Tabelle leer ist.

```
SELECT  *  
FROM    my_employee;
```

21. Verwerfen Sie den letzten DELETE-Vorgang, nicht jedoch den vorhergehenden INSERT-Vorgang.

```
ROLLBACK TO step_17;
```

22. Prüfen Sie, ob die neue Zeile weiterhin intakt ist.

```
SELECT *  
FROM my_employee;
```

23. Schreiben Sie die hinzugefügten Daten dauerhaft fest.

```
COMMIT;
```

Führen Sie die folgende Übung durch, falls Sie noch Zeit haben:

24. Ändern Sie das Skript `lab_10_06.sql` so, dass die `USERID` automatisch generiert wird, indem der erste Buchstabe des Vornamens und die ersten sieben Zeichen des Nachnamens verkettet werden. Für die generierte `USERID` muss Kleinschreibung verwendet werden. Daher sollte das Skript nicht zur Eingabe der `USERID` auffordern. Speichern Sie dieses Skript in der Datei `lab_10_24.sql`.

```
SET ECHO OFF  
SET VERIFY OFF  
INSERT INTO my_employee  
VALUES (&p_id, '&p_last_name', '&p_first_name',  
        lower(substr('&p_first_name', 1, 1) ||  
        substr('&p_last_name', 1, 7)), &p_salary);  
  
SET VERIFY ON  
SET ECHO ON  
UNDEFINE p_first_name  
UNDEFINE p_last_name
```

25. Führen Sie das Skript `lab_10_24.sql` aus, um den folgenden Datensatz einzufügen:

ID	LAST_NAME	FIRST_NAME	USERID	SALARY
6	Anthony	Mark	manthony	1230

26. Prüfen Sie, ob die neue Zeile mit korrekter `USERID` hinzugefügt wurde.

```
SELECT *  
FROM my_employee  
WHERE ID='6';
```

# **Übungen zu Lektion 11 – Tabellen mit DDL- Anweisungen erstellen und verwalten**

## **Kapitel 11**

# Übungen zu Lektion 11 – Überblick

---

## Lektionsüberblick

Diese Übung behandelt folgende Themen:

- Neue Tabellen erstellen
- Neue Tabellen mithilfe der Syntax `CREATE TABLE AS` erstellen
- Vorhandensein von Tabellen prüfen
- Tabellen ändern
- Spalten hinzufügen
- Spalten löschen
- Tabellen in den schreibgeschützten Status setzen
- Tabellen löschen

**Hinweis:** Führen Sie das folgende Skript aus, bevor Sie diese Übung beginnen:

`/home/oracle/labs/sql1/code_ex/cleanup_scripts/cleanup_11.sql`

# Übung 1 zu Lektion 11 – Data Definition Language: Einführung

## Überblick

In dieser Übung erstellen Sie mit der Anweisung `CREATE TABLE` neue Tabellen. Anschließend prüfen Sie, ob die neuen Tabellen der Datenbank hinzugefügt wurden. Darüber hinaus lernen Sie, eine Tabelle in den schreibgeschützten Modus zu setzen und sie anschließend in den Schreibzugriffsmodus zurückzusetzen. Mit dem Befehl `ALTER TABLE` bearbeiten Sie Tabellenspalten.

## Hinweise

- Um die Abfragen in SQL Developer auszuführen, klicken Sie für alle DDL- und DML-Anweisungen auf das Symbol **Run Script** (oder drücken F5). Auf diese Weise werden die Rückmeldungen in der Registerkarte **Script Output** angezeigt. Für `SELECT`-Abfragen verwenden Sie weiterhin das Symbol **Execute Statement** oder drücken F9, um die formatierte Ausgabe in der Registerkarte **Results** anzuzeigen.
- Führen Sie das Skript `cleanup_11.sql` unter `/home/oracle/labs/sql1/code_ex/cleanup_scripts` aus, bevor Sie mit der Durchführung der folgenden Aufgaben beginnen.

## Aufgaben

1. Erstellen Sie die Tabelle `DEPT` auf Basis des folgenden Tabelleninstanzdiagramms. Speichern Sie die Anweisung im Skript `lab_11_01.sql`. Führen Sie anschließend die Skriptanweisung aus, um die Tabelle zu erstellen. Prüfen Sie, ob die Tabelle erstellt wurde.

Column Name	ID	NAME
Key Type	Primary key	
Nulls/Unique		
FK Table		
FK Column		
Data type	NUMBER	VARCHAR2
Length	7	25

```
DESCRIBE dept
Name Null    Type
-----
ID   NOT NULL  NUMBER(7)
NAME                VARCHAR2(25)
```

- Erstellen Sie die Tabelle `EMP` basierend auf dem folgenden Tabelleninstanzdiagramm. Speichern Sie die Anweisung im Skript `lab_11_02.sql`. Führen Sie anschließend die Skriptanweisung aus, um die Tabelle zu erstellen. Prüfen Sie, ob die Tabelle erstellt wurde.

Column Name	ID	LAST_NAME	FIRST_NAME	DEPT_ID
<b>Key Type</b>				
<b>Nulls/Unique</b>				
<b>FK Table</b>				DEPT
<b>FK Column</b>				ID
<b>Data type</b>	NUMBER	VARCHAR2	VARCHAR2	NUMBER
<b>Length</b>	7	25	25	7

```
DESCRIBE emp
Name      Null Type
-----
ID         NUMBER(7)
LAST_NAME  VARCHAR2(25)
FIRST_NAME VARCHAR2(25)
DEPT_ID    NUMBER(7)
```

- Ändern Sie die Tabelle `EMP`. Fügen Sie die Spalte `COMMISSION` vom Datentyp `NUMBER` mit 2 Gesamtstellen und 2 Nachkommastellen hinzu. Prüfen Sie die Änderungen.

```
table EMP altered.
DESCRIBE emp
Name      Null Type
-----
ID         NUMBER(7)
LAST_NAME  VARCHAR2(25)
FIRST_NAME VARCHAR2(25)
DEPT_ID    NUMBER(7)
COMMISSION NUMBER(2,2)
```

- Ändern Sie die Tabelle `EMP` so, dass längere Nachnamen von Mitarbeitern zugelassen werden. Prüfen Sie die Änderung.

```
table EMP altered.
DESCRIBE emp
Name      Null Type
-----
ID         NUMBER(7)
LAST_NAME  VARCHAR2(50)
FIRST_NAME VARCHAR2(25)
DEPT_ID    NUMBER(7)
COMMISSION NUMBER(2,2)
```

5. Löschen Sie die Spalte `FIRST_NAME` aus der Tabelle `EMP`. Prüfen Sie die Änderung, indem Sie die Beschreibung der Tabelle anzeigen.

```
table EMP altered.  
DESCRIBE emp  
Name          Null Type  
-----  
ID             NUMBER(7)  
LAST_NAME      VARCHAR2(50)  
DEPT_ID        NUMBER(7)  
COMMISSION     NUMBER(2,2)
```

6. Markieren Sie in der Tabelle `EMP` die Spalte `DEPT_ID` als `UNUSED`. Prüfen Sie die Änderung, indem Sie die Beschreibung der Tabelle anzeigen.

```
table EMP altered.  
DESCRIBE emp  
Name          Null Type  
-----  
ID             NUMBER(7)  
LAST_NAME      VARCHAR2(50)  
COMMISSION     NUMBER(2,2)
```

7. Löschen Sie alle als `UNUSED` markierten Spalten aus der Tabelle `EMP`.
8. Erstellen Sie die Tabelle `EMPLOYEES2` basierend auf der Struktur der Tabelle `EMPLOYEES`. Nehmen Sie nur die Spalten `EMPLOYEE_ID`, `FIRST_NAME`, `LAST_NAME`, `SALARY` und `DEPARTMENT_ID` auf. Nennen Sie die Spalten in der neuen Tabelle jeweils `ID`, `FIRST_NAME`, `LAST_NAME`, `SALARY` und `DEPT_ID`.

```
describe employees2  
Name          Null      Type  
-----  
ID             NUMBER(6)  
FIRST_NAME     VARCHAR2(20)  
LAST_NAME      NOT NULL  VARCHAR2(25)  
SALARY         NUMBER(8,2)  
DEPT_ID        NUMBER(4)
```

9. Ändern Sie den Status der Tabelle `EMPLOYEES2` in schreibgeschützt.

10. Fügen Sie der Tabelle `EMPLOYEES2` die Spalte `JOB_ID` hinzu.

**Hinweis:** Sie erhalten die Fehlermeldung "Update operation not allowed on table". Sie können der Tabelle keine Spalten hinzufügen, weil sie schreibgeschützt ist.

```
Error starting at line 4 in command:
ALTER TABLE EMPLOYEES2
ADD job_id VARCHAR2(9)
Error report:
SQL Error: ORA-12081: update operation not allowed on table "ORA1"."EMPLOYEES2"
12081. 00000 - "update operation not allowed on table \"%s\".\"%s\""
*Cause:      An attempt was made to update a read-only materialized view.
*Action:     No action required. Only Oracle is allowed to update a
              read-only materialized view.
```

11. Setzen Sie die Tabelle `EMPLOYEES2` in den Modus mit Schreibzugriff zurück. Fügen Sie erneut dieselbe Spalte hinzu.

Nachdem der Tabelle wieder der Status `READ WRITE` zugewiesen wurde, können Sie der Tabelle eine Spalte hinzufügen.

Die folgenden Meldungen werden angezeigt:

```
table EMPLOYEES2 altered.
table EMPLOYEES2 altered.
DESCRIBE employees2
Name          Null          Type
-----
ID             NUMBER(6)
FIRST_NAME     VARCHAR2(20)
LAST_NAME      NOT NULL    VARCHAR2(25)
SALARY         NUMBER(8,2)
DEPT_ID        NUMBER(4)
JOB_ID         VARCHAR2(9)
```

12. Löschen Sie die Tabellen `EMP`, `DEPT` und `EMPLOYEES2`.



## Übung 1 zu Lektion 11 – Lösung: Data Definition Language – Einführung

---

1. Erstellen Sie die Tabelle `DEPT` auf Basis des folgenden Tabelleninstanzdiagramms. Speichern Sie die Anweisung im Skript `lab_11_01.sql`. Führen Sie anschließend die Skriptanweisung aus, um die Tabelle zu erstellen. Prüfen Sie, ob die Tabelle erstellt wurde.

<b>Column Name</b>	ID	NAME
<b>Key Type</b>	Primary key	
<b>Nulls/Unique</b>		
<b>FK Table</b>		
<b>FK Column</b>		
<b>Data type</b>	NUMBER	VARCHAR2
<b>Length</b>	7	25

```
CREATE TABLE dept
(id    NUMBER(7) CONSTRAINT department_id_pk PRIMARY KEY,
 name VARCHAR2(25));
```

Um die Erstellung der Tabelle zu prüfen und deren Struktur anzuzeigen, setzen Sie folgenden Befehl ab:

```
DESCRIBE dept;
```

2. Erstellen Sie die Tabelle `EMP` basierend auf dem folgenden Tabelleninstanzdiagramm. Speichern Sie die Anweisung im Skript `lab_11_02.sql`. Führen Sie die Skriptanweisung anschließend aus, um die Tabelle zu erstellen. Prüfen Sie, ob die Tabelle erstellt wurde.

<b>Column Name</b>	ID	LAST_NAME	FIRST_NAME	DEPT_ID
<b>Key Type</b>				
<b>Nulls/Unique</b>				
<b>FK Table</b>				DEPT
<b>FK Column</b>				ID
<b>Data type</b>	NUMBER	VARCHAR2	VARCHAR2	NUMBER
<b>Length</b>	7	25	25	7

```
CREATE TABLE emp
(id          NUMBER(7),
last_name    VARCHAR2(25),
first_name   VARCHAR2(25),
dept_id      NUMBER(7)
CONSTRAINT emp_dept_id_FK REFERENCES dept (id)
);
```

Um die Erstellung der Tabelle zu prüfen und deren Struktur anzuzeigen, setzen Sie folgenden Befehl ab:

```
DESCRIBE emp
```

3. Ändern Sie die Tabelle EMP. Fügen Sie die Spalte COMMISSION vom Datentyp NUMBER mit 2 Gesamtstellen und 2 Nachkommastellen hinzu. Prüfen Sie die Änderungen.

```
ALTER TABLE emp
ADD commission NUMBER(2,2);

DESCRIBE emp
```

4. Ändern Sie die Tabelle EMP so, dass längere Nachnamen von Mitarbeitern zugelassen werden. Prüfen Sie die Änderung.

```
ALTER TABLE emp
MODIFY (last_name VARCHAR2(50));

DESCRIBE emp
```

5. Löschen Sie die Spalte FIRST\_NAME aus der Tabelle EMP. Prüfen Sie die Änderung, indem Sie die Beschreibung der Tabelle anzeigen.

```
ALTER TABLE emp
DROP COLUMN first_name;

DESCRIBE emp
```

6. Markieren Sie in der Tabelle EMP die Spalte DEPT\_ID als UNUSED. Prüfen Sie die Änderung, indem Sie die Beschreibung der Tabelle anzeigen.

```
ALTER TABLE emp
SET UNUSED (dept_id);

DESCRIBE emp
```

7. Löschen Sie alle als UNUSED markierten Spalten aus der Tabelle EMP.

```
ALTER TABLE emp
DROP UNUSED COLUMNS;
```

8. Erstellen Sie die Tabelle `EMPLOYEES2` basierend auf der Struktur der Tabelle `EMPLOYEES`. Nehmen Sie nur die Spalten `EMPLOYEE_ID`, `FIRST_NAME`, `LAST_NAME`, `SALARY` und `DEPARTMENT_ID` auf. Nennen Sie die Spalten in der neuen Tabelle jeweils `ID`, `FIRST_NAME`, `LAST_NAME`, `SALARY` und `DEPT_ID`. Prüfen Sie, ob die Tabelle erstellt wurde.

```
CREATE TABLE employees2 AS
  SELECT  employee_id id, first_name, last_name, salary,
          department_id dept_id
  FROM    employees;
DESCRIBE employees2
```

9. Ändern Sie den Status der Tabelle `EMPLOYEES2` in schreibgeschützt.

```
ALTER TABLE employees2 READ ONLY;
```

10. Fügen Sie der Tabelle `EMPLOYEES2` die Spalte `JOB_ID` hinzu.

**Hinweis:** Sie erhalten die Fehlermeldung "Update operation not allowed on table". Sie können der Tabelle keine Spalten hinzufügen, weil sie schreibgeschützt ist.

```
ALTER TABLE employees2
ADD job_id VARCHAR2(9);
```

11. Setzen Sie die Tabelle `EMPLOYEES2` in den Modus mit Schreibzugriff zurück. Fügen Sie erneut dieselbe Spalte hinzu.

Nachdem der Tabelle wieder der Status `READ WRITE` zugewiesen wurde, können Sie der Tabelle eine Spalte hinzufügen.

```
ALTER TABLE employees2 READ WRITE;

ALTER TABLE employees2
ADD job_id VARCHAR2(9);
DESCRIBE employees2
```

12. Löschen Sie die Tabellen `EMP`, `DEPT` und `EMPLOYEES2`.

**Hinweis:** Sie können selbst eine Tabelle im Modus `READ ONLY` löschen. Testen Sie dies, indem Sie die Tabelle wieder in den schreibgeschützten Modus versetzen und dann den Befehl `DROP TABLE` absetzen. Die Tabellen werden gelöscht.

```
DROP TABLE emp;
DROP TABLE dept;
DROP TABLE employees2;
```



# **Zusätzliche Übungen und Lösungen**

## **Kapitel 12**

# Übung 1

---

## Übungsüberblick

In diesen Übungen sammeln Sie zusätzliche praktische Erfahrungen zu folgenden Themen:

- Einfache SQL-`SELECT`-Anweisungen
- Einfache SQL Developer-Befehle
- SQL-Funktionen

## Übung 1 – Zusätzliche Übungen

### Überblick

In diesen Übungen können Sie zusätzliche praktische Erfahrung zu den folgenden Themen sammeln: einfache SQL-SELECT-Anweisungen, einfache SQL Developer-Befehle und SQL-Funktionen.

### Aufgaben

1. Die Personalabteilung muss die Daten von allen Mitarbeitern ermitteln, die nach 1997 eingestellt wurden.

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY
1	141	Trenna	Rajs	TRAJS	650.121.8009	17-OCT-03	ST_CLERK	3500
2	142	Curtis	Davies	CDAVIES	650.121.2994	29-JAN-05	ST_CLERK	3100
3	143	Randall	Matos	RMATOS	650.121.2874	15-MAR-06	ST_CLERK	2600
4	144	Peter	Vargas	PVARGAS	650.121.2004	09-JUL-06	ST_CLERK	2500

2. Die Personalabteilung benötigt einen Bericht über Mitarbeiter, die Provisionen erhalten. Zeigen Sie Nachname, Tätigkeits-ID, Gehalt und Provision für diese Mitarbeiter an. Sortieren Sie die Daten in absteigender Reihenfolge nach dem Gehalt.

	LAST_NAME	JOB_ID	SALARY	COMMISSION_PCT
1	Abel	SA_REP	11000	0.3
2	Zlotkey	SA_MAN	10500	0.2
3	Taylor	SA_REP	8600	0.2
4	Grant	SA_REP	7000	0.15

3. Die Personalabteilung benötigt für Budgetierungszwecke einen Bericht über geplante Gehaltserhöhungen. Der Bericht soll die Mitarbeiter anzeigen, die nicht provisionsberechtigt sind, deren Gehalt aber um 10 % erhöht wird. (Runden Sie die Gehälter.)

	New salary
1	The salary of Whalen after a 10% raise is 4840
2	The salary of Hartstein after a 10% raise is 14300
3	The salary of Fay after a 10% raise is 6600
4	The salary of Higgins after a 10% raise is 13209
5	The salary of Gietz after a 10% raise is 9130
6	The salary of King after a 10% raise is 26400
7	The salary of Kochhar after a 10% raise is 18700
8	The salary of De Haan after a 10% raise is 18700
9	The salary of Hunold after a 10% raise is 9900
10	The salary of Ernst after a 10% raise is 6600
11	The salary of Lorentz after a 10% raise is 4620
12	The salary of Mourgos after a 10% raise is 6380
13	The salary of Rajs after a 10% raise is 3850
14	The salary of Davies after a 10% raise is 3410
15	The salary of Matos after a 10% raise is 2860
16	The salary of Vargas after a 10% raise is 2750

4. Erstellen Sie einen Bericht über die Mitarbeiter und die Dauer ihrer Beschäftigung. Zeigen Sie die Nachnamen aller Mitarbeiter zusammen mit der Betriebszugehörigkeit in Jahren und ganzen Monaten an. Sortieren Sie den Bericht nach der Dauer der Betriebszugehörigkeit. Der Mitarbeiter mit der längsten Betriebszugehörigkeit soll oben in der Liste stehen.

	LAST_NAME	YEARS	MONTHS
3	Higgins	11	11
4	King	10	11
5	Whalen	10	8
6	Rajs	10	7
7	Hartstein	10	3
8	Abel	10	0
9	Davies	9	4
10	Fay	8	9
11	Kochhar	8	8
12	Hunold	8	5
13	Taylor	8	2
14	Matos	8	2
15	Vargas	7	10
16	Lorentz	7	3
17	Grant	7	0
18	Ernst	7	0
19	Mourgos	6	6
20	Zlotkey	6	4

5. Zeigen Sie die Mitarbeiter an, deren Nachname mit "J", "K", "L" oder "M" beginnt.

	LAST_NAME
1	King
2	Kochhar
3	Lorentz
4	Matos
5	Mourgos



6. Erstellen Sie einen Bericht, der alle Mitarbeiter anzeigt und mit Yes oder No angibt, ob sie provisionsberechtigt sind. Verwenden Sie in Ihrer Abfrage den Ausdruck `DECODE`.

	LAST_NAME	SALARY	COMMISSION
1	Whalen	4400	No
2	Hartstein	13000	No
3	Fay	6000	No
4	Higgins	12008	No
5	Gietz	8300	No
6	King	24000	No
7	Kochhar	17000	No
8	De Haan	17000	No
9	Hunold	9000	No
10	Ernst	6000	No
11	Lorentz	4200	No
12	Mourgos	5800	No
13	Rajs	3500	No
14	Davies	3100	No
15	Matos	2600	No
16	Vargas	2500	No
17	Zlotkey	10500	Yes
18	Abel	11000	Yes
19	Taylor	8600	Yes
20	Grant	7000	Yes

Bei diesen Übungen können Sie zusätzliche praktische Erfahrung zu den folgenden Themen sammeln: einfache SQL-`SELECT`-Anweisungen, einfache SQL Developer-Befehle, SQL-Funktionen, Joins und Gruppenfunktionen.

7. Erstellen Sie einen Bericht, der Abteilungsname, Standort-ID, Nachname, Tätigkeit und Gehalt der Mitarbeiter an einem bestimmten Standort anzeigt. Erstellen Sie eine Eingabeaufforderung für den Benutzer, um den Standort einzugeben. Beispiel: Wenn der Benutzer 1800 eingibt, sieht das Ergebnis wie folgt aus:

	DEPARTMENT_NAME	LOCATION_ID	LAST_NAME	JOB_ID	SALARY
1	Marketing	1800	Hartstein	MK_MAN	13000
2	Marketing	1800	Fay	MK_REP	6000

8. Ermitteln Sie die Anzahl der Mitarbeiter, deren Nachname mit dem Buchstaben "n" endet. Erstellen Sie zwei mögliche Lösungen.

	COUNT(*)
1	3

9. Erstellen Sie einen Bericht, der Name, Standort und Anzahl der Mitarbeiter pro Abteilung anzeigt. Stellen Sie sicher, dass der Bericht auch Abteilungen ohne Mitarbeiter enthält.

	DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID	COUNT(E.EMPLOYEE_ID)
1	80	Sales	2500	3
2	110	Accounting	1700	2
3	60	IT	1400	3
4	10	Administration	1700	1
5	90	Executive	1700	3
6	20	Marketing	1800	2
7	50	Shipping	1500	5
8	190	Contracting	1700	0

10. Die Personalabteilung muss die Tätigkeiten in den Abteilungen 10 und 20 ermitteln. Erstellen Sie einen Bericht, der die Tätigkeit-IDs (JOB\_ID) für diese Abteilungen anzeigt.

	JOB_ID
1	AD_ASST
2	MK_MAN
3	MK_REP

11. Erstellen Sie einen Bericht, der die Tätigkeiten anzeigt, die in den Abteilungen Administration und Executive ermittelt wurden. Zeigen Sie auch die Anzahl der Mitarbeiter für diese Tätigkeiten an. Die Tätigkeit, die von den meisten Mitarbeitern ausgeführt wird, soll an erster Stelle aufgeführt werden.

	JOB_ID	FREQUENCY
1	AD_VP	2
2	AD_PREP	1
3	AD_ASST	1

In diesen Übungen können Sie zusätzliche praktische Erfahrung zu den folgenden Themen sammeln: einfache SQL-SELECT-Anweisungen, einfache SQL Developer-Befehle, SQL-Funktionen, Joins, Gruppenfunktionen und Unterabfragen.

12. Zeigen Sie alle Mitarbeiter an, die in der ersten Monathälfte (vor dem 16. des Monats) eingestellt wurden – unabhängig vom jeweiligen Jahr.

	LAST_NAME	HIRE_DATE
1	De Haan	13-JAN-01
2	Hunold	03-JAN-06
3	Lorentz	07-FEB-07
4	Matos	15-MAR-06
5	Vargas	09-JUL-06
6	Abel	11-MAY-04
7	Higgins	07-JUN-02
8	Gietz	07-JUN-02

13. Erstellen Sie einen Bericht, der für jeden Mitarbeiter folgende Informationen anzeigt: Nachname, Gehalt und Gehalt in Tausend Dollar.

	A Z LAST_NAME	A Z SALARY	A Z THOUSANDS
1	King	24000	24
2	Kochhar	17000	17
3	De Haan	17000	17
4	Hunold	9000	9
5	Ernst	6000	6
6	Lorentz	4200	4
7	Mourgos	5800	5
8	Rajs	3500	3
9	Davies	3100	3
10	Matos	2600	2
11	Vargas	2500	2
12	Zlotkey	10500	10
13	Abel	11000	11
14	Taylor	8600	8
15	Grant	7000	7
16	Whalen	4400	4
17	Hartstein	13000	13
18	Fay	6000	6
19	Higgins	12008	12
20	Gietz	8300	8

14. Zeigen Sie alle Mitarbeiter an, deren Manager ein Gehalt von mehr als \$ 15.000 beziehen. Zeigen Sie folgende Daten an: Name des Mitarbeiters, Name des Managers, Gehalt und Gehaltsstufe des Managers.

	A Z LAST_NAME	A Z MANAGER	A Z SALARY	A Z GRADE_LEVEL
1	Kochhar	King	24000	E
2	De Haan	King	24000	E
3	Mourgos	King	24000	E
4	Zlotkey	King	24000	E
5	Hartstein	King	24000	E
6	Whalen	Kochhar	17000	E
7	Higgins	Kochhar	17000	E
8	Hunold	De Haan	17000	E

15. Zeigen Sie Abteilungsnummer, Name, Anzahl der Mitarbeiter und Durchschnittsgehalt für alle Abteilungen an, zusammen mit den Namen, Gehältern und Tätigkeiten der Mitarbeiter in jeder Abteilung.

	DEPARTMENT_ID	DEPARTMENT_NAME	EMPLOYEES	AVG_SAL	LAST_NAME	SALARY	JOB_ID
1	10	Administration	1	4400.00	Whalen	4400	AD_ASST
2	20	Marketing	2	9500.00	Hartstein	13000	MK_MAN
3	20	Marketing	2	9500.00	Fay	6000	MK_REP
4	50	Shipping	5	3500.00	Davies	3100	ST_CLERK
5	50	Shipping	5	3500.00	Matos	2600	ST_CLERK
6	50	Shipping	5	3500.00	Rajs	3500	ST_CLERK
7	50	Shipping	5	3500.00	Mourgos	5800	ST_MAN
8	50	Shipping	5	3500.00	Vargas	2500	ST_CLERK
9	60	IT	3	6400.00	Hunold	9000	IT_PROG
10	60	IT	3	6400.00	Lorentz	4200	IT_PROG
11	60	IT	3	6400.00	Ernst	6000	IT_PROG
12	80	Sales	3	10033.33	Zlotkey	10500	SA_MAN
13	80	Sales	3	10033.33	Abel	11000	SA_REP
14	80	Sales	3	10033.33	Taylor	8600	SA_REP
15	90	Executive	3	19333.33	Kochhar	17000	AD_VP
16	90	Executive	3	19333.33	King	24000	AD_PRES
17	90	Executive	3	19333.33	De Haan	17000	AD_VP
18	110	Accounting	2	10154.00	Gietz	8300	AC_ACCOUNT
19	110	Accounting	2	10154.00	Higgins	12008	AC_MGR
20	(null)	(null)	0	No average	Grant	7000	SA_REP

16. Erstellen Sie einen Bericht, um die Abteilungsnummer und das niedrigste Gehalt für die Abteilung mit dem höchsten Durchschnittsgehalt anzuzeigen.

	DEPARTMENT_ID	MIN(SALARY)
1	90	17000

17. Erstellen Sie einen Bericht, der die Abteilungen anzeigt, in denen keine Sales Representatives arbeiten. Die Ausgabe soll die Abteilungsnummer, den Abteilungsnamen, die Manager-ID und den Standort enthalten.

	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	50	Shipping	124	1500
2	60	IT	103	1400
3	110	Accounting	205	1700
4	20	Marketing	201	1800
5	10	Administration	200	1700
6	190	Contracting	(null)	1700
7	90	Executive	100	1700

- a. Weniger als 3 Mitarbeiter:

b. Höchste Anzahl von Mitarbeitern:

c. Niedrigste Anzahl von Mitarbeitern:

19. Erstellen Sie einen Bericht, der Personalnummer, Nachname, Gehalt, Abteilungsnummer und Durchschnittsgehalt der Abteilung für alle Mitarbeiter anzeigt.

[illegible]

20. Erstellen Sie eine Jahrestagsübersicht nach dem Einstellungsdatum der Mitarbeiter. Sortieren Sie die Jahrestage in aufsteigender Reihenfolge.

	LAST_NAME	BIRTHDAY
1	Hunold	January 03
2	De Haan	January 13
3	Davies	January 29
4	Zlotkey	January 29
5	Lorentz	February 07
6	Hartstein	February 17
7	Matos	March 15
8	Taylor	March 24
9	Abel	May 11
10	Ernst	May 21
11	Grant	May 24
12	Higgins	June 07
13	Gietz	June 07
14	King	June 17
15	Vargas	July 09
16	Fay	August 17
17	Whalen	September 17
18	Kochhar	September 21
19	Rajs	October 17
20	Mourgos	November 16

# Übung 1 – Lösungen: Zusätzliche Übungen

---

## Überblick

Im Folgenden sind die Lösungen für Übung 1 angegeben.

## Aufgaben

1. Die Personalabteilung muss die Daten von allen Mitarbeitern ermitteln, die nach 1997 eingestellt wurden.

```
SELECT *
FROM   employees
WHERE  job_id = 'ST_CLERK'
AND    hire_date > '31-DEC-1997';
```

2. Die Personalabteilung benötigt einen Bericht über Mitarbeiter, die Provisionen erhalten. Zeigen Sie Nachname, Tätigkeits-ID, Gehalt und Provision für diese Mitarbeiter an. Sortieren Sie die Daten in absteigender Reihenfolge nach dem Gehalt.

```
SELECT last_name, job_id, salary, commission_pct
FROM   employees
WHERE  commission_pct IS NOT NULL
ORDER BY salary DESC;
```

3. Die Personalabteilung benötigt für Budgetierungszwecke einen Bericht über geplante Gehaltserhöhungen. Der Bericht soll die Mitarbeiter anzeigen, die nicht provisionsberechtigt sind, deren Gehalt aber um 10 % erhöht wird. (Runden Sie die Gehälter.)

```
SELECT 'The salary of '||last_name||' after a 10% raise is '
      || ROUND(salary*1.10) "New salary"
FROM   employees
WHERE  commission_pct IS NULL;
```

4. Erstellen Sie einen Bericht über die Mitarbeiter und die Dauer ihrer Beschäftigung. Zeigen Sie die Nachnamen aller Mitarbeiter zusammen mit der Betriebszugehörigkeit in Jahren und ganzen Monaten an. Sortieren Sie den Bericht nach der Dauer der Betriebszugehörigkeit. Der Mitarbeiter mit der längsten Betriebszugehörigkeit soll oben in der Liste stehen.

```
SELECT last_name,
       TRUNC(MONTHS_BETWEEN(SYSDATE, hire_date) / 12) YEARS,
       TRUNC(MOD(MONTHS_BETWEEN(SYSDATE, hire_date), 12))
       MONTHS
FROM   employees
ORDER BY years DESC, MONTHS desc;
```

5. Zeigen Sie die Mitarbeiter an, deren Nachname mit "J", "K", "L" oder "M" beginnt.

```
SELECT last_name
FROM   employees
WHERE  SUBSTR(last_name, 1,1) IN ('J', 'K', 'L', 'M');
```

6. Erstellen Sie einen Bericht, der alle Mitarbeiter anzeigt und mit Yes oder No angibt, ob sie provisionsberechtigt sind. Verwenden Sie in Ihrer Abfrage den Ausdruck `DECODE`.

```
SELECT last_name, salary,
       decode(commission_pct, NULL, 'No', 'Yes') commission
FROM   employees;
```

Bei diesen Übungen können Sie zusätzliche praktische Erfahrung zu den folgenden Themen sammeln: einfache SQL-SELECT-Anweisungen, einfache SQL Developer-Befehle, SQL-Funktionen, Joins und Gruppenfunktionen.

7. Erstellen Sie einen Bericht, der Abteilungsname, Standort-ID, Nachname, Tätigkeit und Gehalt der Mitarbeiter an einem bestimmten Standort anzeigt. Erstellen Sie eine Eingabeaufforderung für den Benutzer, um den Standort einzugeben.

Wenn Sie dazu aufgefordert werden, geben Sie für `location_id` den Wert 1800 ein.

```
SELECT d.department_name, d.location_id, e.last_name, e.job_id,
       e.salary
FROM   employees e JOIN departments d
ON     e.department_id = d.department_id
AND    d.location_id = &location_id;
```

8. Ermitteln Sie die Anzahl der Mitarbeiter, deren Nachname mit dem Buchstaben "n" endet. Erstellen Sie zwei mögliche Lösungen.

```
SELECT COUNT(*)
FROM   employees
WHERE  last_name LIKE '%n';
--oder
SELECT COUNT(*)
FROM   employees
WHERE  SUBSTR(last_name, -1) = 'n';
```

9. Erstellen Sie einen Bericht, der Name, Standort und Anzahl der Mitarbeiter pro Abteilung anzeigt. Stellen Sie sicher, dass der Bericht auch Abteilungen ohne Mitarbeiter enthält.

```
SELECT d.department_id, d.department_name,
       d.location_id, COUNT(e.employee_id)
FROM   employees e RIGHT OUTER JOIN departments d
ON     e.department_id = d.department_id
GROUP BY d.department_id, d.department_name, d.location_id;
```

10. Die Personalabteilung muss die Tätigkeiten in den Abteilungen 10 und 20 ermitteln. Erstellen Sie einen Bericht, der die Tätigkeits-IDs (`JOB_ID`) für diese Abteilungen anzeigt.

```
SELECT DISTINCT job_id
FROM   employees
WHERE  department_id IN (10, 20);
```



11. Erstellen Sie einen Bericht, der die Tätigkeiten anzeigt, die in den Abteilungen Administration und Executive ermittelt wurden. Zeigen Sie auch die Anzahl der Mitarbeiter für diese Tätigkeiten an. Die Tätigkeit, die von den meisten Mitarbeitern ausgeführt wird, soll an erster Stelle aufgeführt werden.

```
SELECT e.job_id, count(e.job_id) FREQUENCY
FROM   employees e JOIN departments d
ON     e.department_id = d.department_id
WHERE  d.department_name IN ('Administration', 'Executive')
GROUP BY e.job_id
ORDER BY FREQUENCY DESC;
```

Bei diesen Übungen können Sie zusätzliche praktische Erfahrung zu den folgenden Themen sammeln: einfache SQL-SELECT-Anweisungen, einfache SQL Developer-Befehle, SQL-Funktionen, Joins, Gruppenfunktionen und Unterabfragen.

12. Zeigen Sie alle Mitarbeiter an, die in der ersten Monatshälfte (vor dem 16. des Monats) eingestellt wurden – unabhängig vom jeweiligen Jahr.

```
SELECT last_name, hire_date
FROM   employees
WHERE  TO_CHAR(hire_date, 'DD') < 16;
```

13. Erstellen Sie einen Bericht, der für jeden Mitarbeiter folgende Informationen anzeigt: Nachname, Gehalt und Gehalt in Tausend Dollar.

```
SELECT last_name, salary, TRUNC(salary, -3)/1000 Thousands
FROM   employees;
```

14. Zeigen Sie alle Mitarbeiter an, deren Manager ein Gehalt von mehr als \$ 15.000 beziehen. Zeigen Sie folgende Daten an: Name des Mitarbeiters, Name des Managers, Gehalt und Gehaltsstufe des Managers.

```
SELECT e.last_name, m.last_name manager, m.salary,
j.grade_level
FROM   employees e JOIN employees m
ON     e.manager_id = m.employee_id
JOIN   job_grades j
ON     m.salary BETWEEN j.lowest_sal AND j.highest_sal
AND    m.salary > 15000;
```

15. Zeigen Sie Abteilungsnummer, Name, Anzahl der Mitarbeiter und Durchschnittsgehalt für alle Abteilungen an, zusammen mit den Namen, Gehältern und Tätigkeiten der Mitarbeiter in jeder Abteilung.

```
SELECT d.department_id, d.department_name,
count(e1.employee_id) employees,
NVL(TO_CHAR(AVG(e1.salary), '99999.99'), 'No average' )
avg_sal,
e2.last_name, e2.salary, e2.job_id
```

```

FROM    departments d RIGHT OUTER JOIN employees e1
ON      d.department_id = e1.department_id
RIGHT OUTER JOIN employees e2
ON      d.department_id = e2.department_id
GROUP BY d.department_id, d.department_name, e2.last_name,
        e2.salary,
        e2.job_id
ORDER BY d.department_id, employees;

```

16. Erstellen Sie einen Bericht, um die Abteilungsnummer und das niedrigste Gehalt für die Abteilung mit dem höchsten Durchschnittsgehalt anzuzeigen.

```

SELECT department_id, MIN(salary)
FROM    employees
GROUP BY department_id
HAVING AVG(salary) = (SELECT MAX(AVG(salary))
                     FROM    employees
                     GROUP BY department_id);

```

17. Erstellen Sie einen Bericht, der die Abteilungen anzeigt, in denen keine Sales Representatives arbeiten. Die Ausgabe soll die Abteilungsnummer, den Abteilungsnamen, die Manager-ID und den Standort enthalten.

```

SELECT *
FROM    departments
WHERE   department_id NOT IN(SELECT department_id
                             FROM employees
                             WHERE job_id = 'SA_REP'
                             AND department_id IS NOT NULL);

```

18. Erstellen Sie die folgenden Statistikberichte für die Personalabteilung. Nehmen Sie Abteilungsnummer, Abteilungsname und Anzahl der Mitarbeiter für jede Abteilung auf, die folgende Bedingungen erfüllt:

- a. Weniger als 3 Mitarbeiter:

```

SELECT d.department_id, d.department_name, COUNT(*)
FROM    departments d JOIN employees e
ON      d.department_id = e.department_id
GROUP BY d.department_id, d.department_name
HAVING COUNT(*) < 3;

```

b. Höchste Anzahl von Mitarbeitern:

```
SELECT d.department_id, d.department_name, COUNT(*)
FROM   departments d JOIN employees e
ON     d.department_id = e.department_id
GROUP BY d.department_id, d.department_name
HAVING COUNT(*) = (SELECT MAX(COUNT(*))
                   FROM   employees
                   GROUP BY department_id);
```

c. Niedrigste Anzahl von Mitarbeitern:

```
SELECT d.department_id, d.department_name, COUNT(*)
FROM   departments d JOIN employees e
ON     d.department_id = e.department_id
GROUP BY d.department_id, d.department_name
HAVING COUNT(*) = (SELECT MIN(COUNT(*))
                   FROM   employees
                   GROUP BY department_id);
```

19. Erstellen Sie einen Bericht, der Personalnummer, Nachname, Gehalt, Abteilungsnummer und Durchschnittsgehalt der Abteilung für alle Mitarbeiter anzeigt.

```
SELECT e.employee_id, e.last_name, e.department_id, e.salary,
       AVG(s.salary)
FROM   employees e JOIN employees s
ON     e.department_id = s.department_id
GROUP BY e.employee_id, e.last_name, e.department_id,
       e.salary;
```

20. Erstellen Sie eine Jahrestagsübersicht nach dem Einstellungsdatum der Mitarbeiter. Sortieren Sie die Jahrestage in aufsteigender Reihenfolge.

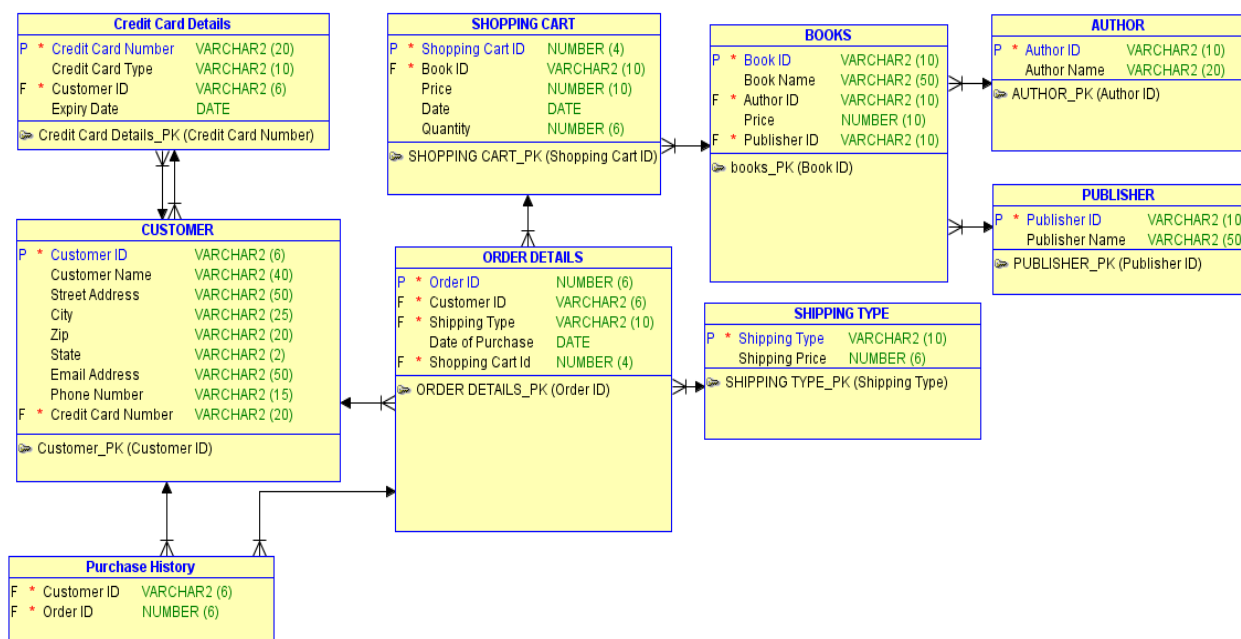
```
SELECT last_name, TO_CHAR(hire_date, 'Month DD') BIRTHDAY
FROM   employees
ORDER BY TO_CHAR(hire_date, 'DDD');
```

# Fallstudie: Onlinebuchhandlung

## Überblick

In dieser Fallstudie erstellen Sie mehrere Datenbanktabellen für eine Onlinebuchhandlung (E-Commerce-Warenkorb). Nach der Erstellung der Tabellen fügen Sie Datensätze in die Datenbank der Buchhandlung ein. Sie aktualisieren und löschen Datensätze und generieren einen Bericht. Die Datenbank enthält nur die wichtigsten Tabellen.

Die folgende Abbildung führt die Tabellen und Spalten auf, die für die Onlinebuchhandlung benötigt werden:



**Hinweis:** Um die Tabellen zu erstellen, führen Sie in SQL Developer die Befehle im Skript `Online_Book_Store_Create_Table.sql` aus. Um die Tabellen zu löschen, führen Sie in SQL Developer die Befehle im Skript `Online_Book_Store_Create_Table.sql` aus. Führen Sie anschließend die Befehle im Skript `<<Online_Book_Store_Populate.sql>>` in SQL Developer aus, um die Tabellen zu erstellen und mit Daten zu füllen.

Sie finden die drei SQL-Skripte im Ordner `/home/oracle/labs/sql1/labs`.

- Wenn Sie die Tabellen mit dem Skript `Online_Book_Store_Create_Table.sql` erstellen, beginnen Sie mit dem 2. Schritt.
- Wenn Sie die Tabellen mit dem Skript `Online_Book_Store_Drop_Table.sql` entfernen, beginnen Sie mit dem 1. Schritt.
- Wenn Sie die Tabellen mit dem Skript `Online_Book_Store_Populate.sql` erstellen und mit Daten füllen, beginnen Sie mit dem 6. Schritt.

## Übung 2

---

### Überblick

In dieser Übung erstellen Sie die Tabellen auf der Basis der folgenden Tabelleninstanz-diagramme. Wählen Sie die geeigneten Datentypen, und fügen Sie Integritäts-Constraints hinzu.

### Aufgaben

#### 1. Tabellendetails

a. Tabellenname: AUTHOR

Spalte	Datentyp	Schlüssel	Tabellenabhängiger Typ
Author_ID	VARCHAR2	PK	
Author_Name	VARCHAR2		

b. Tabellenname: BOOKS

Spalte	Datentyp	Schlüssel	Abhängig von
Book_ID	VARCHAR2	PK	
Book_Name	VARCHAR2		
Author_ID	VARCHAR2	FK	AUTHORS
Price	NUMBER		
Publisher_ID	VARCHAR2	FK	PUBLISHER

c. Tabellenname: CUSTOMER

Spaltenname	Datentyp	Schlüssel	Abhängig von
Customer_ID	VARCHAR2	PK	
Customer_Name	VARCHAR2		
Street_Address	VARCHAR2		
City	VARCHAR2		
Phone_Number	VARCHAR2		
Credit_Card_Number	VARCHAR2	FK	Credit_Card_Details

d. Tabellenname: CREDIT\_CARD\_DETAILS

Spalte	Datentyp	Schlüssel	Abhängig von
Credit_Card_Number	VARCHAR2	PK	
Credit_Card_Type	VARCHAR2		
Expiry_Date	DATE		

e. Tabellenname: ORDER\_DETAILS

Spalte	Datentyp	Schlüssel	Abhängig von
Order_ID	NUMBER	PK	
Customer_ID	VARCHAR2	FK	CUSTOMER
Shipping_Type	VARCHAR2	FK	SHIPPING_TYPE
Date_of_Purchase	DATE		
Shopping_Cart_ID	NUMBER	FK	SHOPPING_CART

f. Tabellenname: PUBLISHER

Spalte	Datentyp	Schlüssel	Tabellenabhängiger Typ
Publisher_ID	VARCHAR2	PK	
Publisher_Name	VARCHAR2		

g. Tabellenname: PURCHASE\_HISTORY

Spalte	Datentyp	Schlüssel	Tabellenabhängiger Typ
Customer_ID	VARCHAR2	FK	CUSTOMER
Order_ID	NUMBER	FK	ORDER_DETAILS

h. Tabellenname: SHIPPING\_TYPE

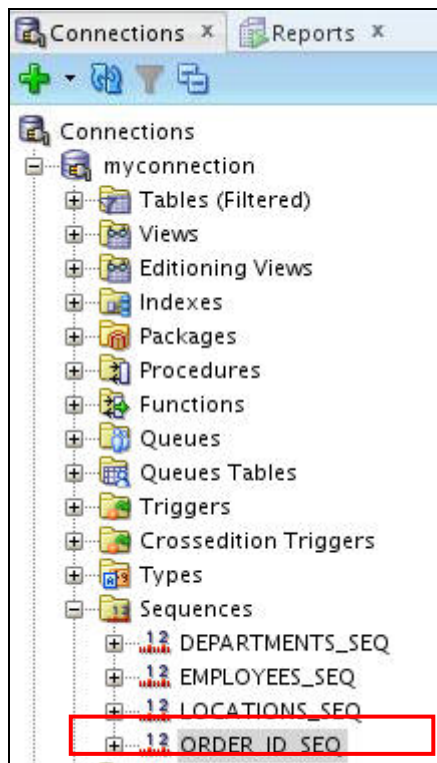
Spalte	Datentyp	Schlüssel	Tabellenabhängiger Typ
Shipping_Type	VARCHAR2	PK	
Shipping_Price	NUMBER		

i. Tabellenname: SHOPPING\_CART

Spalte	Datentyp	Schlüssel	Abhängig von
Shopping_Cart_ID	NUMBER	PK	
Book_ID	VARCHAR2	FK	BOOKS
Price	NUMBER		
Date	DATE		
Quantity	NUMBER		

2. Fügen Sie den erstellten Tabellen zusätzliche referenzielle Integritäts-Constraints hinzu.
3. Prüfen Sie im Connections Navigator von SQL Developer, ob die Tabellen korrekt erstellt wurden.
4. Erstellen Sie eine Sequence, um die einzelnen Zeilen in der Tabelle ORDER\_DETAILS eindeutig zu kennzeichnen.
  - a. Beginnen Sie mit 100, wobei keine Werte im Cache gespeichert werden sollen. Geben Sie der Sequence den Namen ORDER\_ID\_SEQ.

- b. Überprüfen Sie die Existenz der Sequences im Connections Navigator von SQL Developer.



5. Geben Sie Daten in die Tabellen ein. Erstellen Sie ein Skript für jedes Dataset, das hinzugefügt werden soll.

Geben Sie Daten in die folgenden Tabellen ein.

- a. AUTHOR
- b. PUBLISHER
- c. SHIPPING\_TYPE
- d. CUSTOMER
- e. CREDIT\_CARD\_DETAILS
- f. BOOKS
- g. SHOPPING\_CART
- h. ORDER\_DETAILS
- i. PURCHASE\_HISTORY

**Hinweis:** Speichern Sie die Skripte anhand der Aufgabennummer. Speichern Sie beispielsweise das für die Tabelle BOOKS erstellte Skript als labs\_apcs\_5a\_1.sql. Prüfen Sie, ob die Skripte im Ordner /home/oracle/labs gespeichert wurden.

6. Erstellen Sie die View CUSTOMER\_DETAILS, in der Name und Adresse der Kunden sowie detaillierte Informationen zur Bestellung des jeweiligen Kunden angezeigt werden. Sortieren Sie die Ergebnisse nach der Kunden-ID.

	CUSTOMER_NAME	STREET_ADDRESS	ORDER_ID	CUSTOMER_ID	SHIPPING_TYPE	DATE_OF_PURCHASE	SHOPPING_CART_ID
1	VelasquezCarmen	283 King Street	0D0001	CN0001	USPS	12-JUN-01	SC0002
2	Ngao LaDoris	5 Modrany	0D0002	CN0002	USPS	28-JUN-05	SC0005
3	Nagayama Midori	68 Via Centrale	0D0003	CN0003	FedEx	31-JUL-05	SC0007
4	Quick-To-See Mark	6921 King Way	0D0004	CN0004	FedEx	14-AUG-06	SC0004
5	Ropeburn Audry	86 Chu Street	0D0005	CN0005	FedEx	21-SEP-06	SC0003
6	Urguhart Molly	3035 Laurier Blvd.	0D0006	CN0006	DHL	28-OCT-07	SC0001
7	Menchu Roberta	Boulevard de Waterloo 41	0D0007	CN0007	DHL	11-AUG-07	SC0006
8	Biri Ben	398 High St.	0D0008	CN0008	DHL	18-SEP-09	SC0008
9	Catchpole Antoinette	88 Alfred St.	0D0009	CN0009	USPS	25-NOV-09	SC0009

7. Ändern Sie Daten in den Tabellen.

- a. Fügen Sie Details zu einem neuen Buch hinzu. Prüfen Sie, ob die Angabe zum Autor des Buchs in der Tabelle AUTHOR verfügbar ist. Erstellen Sie gegebenenfalls einen Eintrag in der Tabelle AUTHOR.

	BOOK_ID	BOOK_NAME	AUTHOR_ID	PRICE	PUBLISHER_ID
1	BN0001	Florentine Tragedy	AN0002	150	PN0002
2	BN0002	A Vision	AN0002	100	PN0003
3	BN0003	Citizen of the World	AN0001	100	PN0001
4	BN0004	The Complete Poetical Works of Oliver Goldsmith	AN0001	300	PN0001
5	BN0005	Androcles and the Lion	AN0003	90	PN0004
6	BN0006	An Unsocial Socialist	AN0003	80	PN0004
7	BN0007	A Thing of Beauty is a Joy Forever	AN0007	100	PN0002
8	BN0008	Beyond the Pale	AN0008	75	PN0005
9	BN0009	The Clicking of Cuthbert	AN0009	175	PN0005
10	BN0010	Bride of Frankenstein	AN0006	200	PN0001
11	BN0011	Shelley Poetry and Prose	AN0005	150	PN0003
12	BN0012	War and Peace	AN0004	150	PN0002
13	BN0013	Two States	AN0009	150	PN0005

- b. Geben Sie Warenkorbdetails für das Buch ein, das Sie gerade im Schritt 7(a) angelegt haben.

	SHOPPING_CART_ID	BOOK_ID	PRICE	SHOPPING_CART_DATE	QUANTITY
1	SC0001	BN0002	200	12-JUN-01	10
2	SC0002	BN0003	90	31-JUL-05	8
3	SC0003	BN0003	175	28-JUN-05	7
4	SC0004	BN0001	80	14-AUG-06	9
5	SC0005	BN0001	175	21-SEP-06	4
6	SC0006	BN0004	100	11-AUG-07	6
7	SC0007	BN0005	200	28-OCT-07	5
8	SC0008	BN0006	100	25-NOV-09	7
9	SC0009	BN0006	150	18-SEP-09	8
10	SC0010	BN0013	200	12-JUN-06	12



8. Erstellen Sie einen Bericht, der die Kaufhistorie der einzelnen Kunden enthält. Nehmen Sie Kundenname und -ID, Buch-ID, Kaufdatum und Warenkorb-ID in den Bericht auf. Speichern Sie die Befehle für die Generierung des Berichts in der Skriptdatei lab\_apcs\_8.sql.

**Hinweis:** Ihre Ergebnisse können abweichen.

R	CUSTOMER	R	CUSTOMER_ID	R	SHOPPING_CART_ID	R	BOOK_ID	R	DATE_OF_PURCHASE
1	VelasquezCarmen		CN0001		SC0002		BN0003		12-JUN-01
2	Ngao LaDoris		CN0002		SC0005		BN0001		28-JUN-05
3	Nagayama Midori		CN0003		SC0007		BN0005		31-JUL-05
4	Quick-To-See Mark		CN0004		SC0004		BN0001		14-AUG-06
5	Ropeburn Audry		CN0005		SC0003		BN0003		21-SEP-06
6	Urguhart Molly		CN0006		SC0001		BN0002		28-OCT-07
7	Menchu Roberta		CN0007		SC0006		BN0004		11-AUG-07
8	Biri Ben		CN0008		SC0008		BN0006		18-SEP-09
9	Catchpole Antoinette		CN0009		SC0009		BN0006		25-NOV-09

## Übung 2 – Lösungen

---

### Überblick

Im Folgenden sind die Lösungen für Übung 2 angegeben.

### Aufgaben

#### 1. Tabellendetails

##### a. AUTHOR

```
CREATE TABLE AUTHOR
(
    Author_ID VARCHAR2 (10) NOT NULL ,
    Author_Name VARCHAR2 (20)
)
;

COMMENT ON TABLE AUTHOR IS 'Author'
;

ALTER TABLE AUTHOR
    ADD CONSTRAINT AUTHOR_PK PRIMARY KEY (Author_ID);
```

##### b. BOOKS

```
CREATE TABLE BOOKS
(
    Book_ID VARCHAR2 (10) NOT NULL ,
    Book_Name VARCHAR2 (50) ,
    Author_ID VARCHAR2 (10) NOT NULL ,
    Price NUMBER (10) ,
    Publisher_ID VARCHAR2 (10) NOT NULL
)
;

COMMENT ON TABLE BOOKS IS 'Books'
;

ALTER TABLE BOOKS
    ADD CONSTRAINT books_PK PRIMARY KEY ( Book_ID );
```

c. CUSTOMER

```
CREATE TABLE CUSTOMER
(
    Customer_ID VARCHAR2 (6)  NOT NULL ,
    Customer_Name VARCHAR2 (40) ,
    Street_Address VARCHAR2 (50) ,
    City VARCHAR2 (25) ,
    Phone_Number VARCHAR2 (15) ,
    Credit_Card_Number VARCHAR2 (20)  NOT NULL
)
;

COMMENT ON TABLE CUSTOMER IS 'Customer'
;

ALTER TABLE CUSTOMER
    ADD CONSTRAINT Customer_PK PRIMARY KEY ( Customer_ID ) ;
```

d. CREDIT\_CARD\_DETAILS

```
CREATE TABLE CREDIT_CARD_DETAILS
(
    Credit_Card_Number VARCHAR2 (20)  NOT NULL ,
    Credit_Card_Type VARCHAR2 (10) ,
    Expiry_Date DATE
)
;

COMMENT ON TABLE CREDIT_CARD_DETAILS IS 'Credit Card Details'
;

ALTER TABLE CREDIT_CARD_DETAILS
    ADD CONSTRAINT Credit_Card_Details_PK PRIMARY KEY (
Credit_Card_Number) ;
```

e. ORDER\_DETAILS

```
CREATE TABLE ORDER_DETAILS
(
  Order_ID VARCHAR2 (6)  NOT NULL ,
  Customer_ID VARCHAR2 (6)  NOT NULL ,
  Shipping_Type VARCHAR2 (10)  NOT NULL ,
  Date_of_Purchase DATE ,
  Shopping_Cart_ID varchar2(6)  NOT NULL
)
;

COMMENT ON TABLE ORDER_DETAILS IS 'Order Details'
;

ALTER TABLE ORDER_DETAILS
  ADD CONSTRAINT ORDER_DETAILS_PK PRIMARY KEY (Order_ID ) ;
```

f. PUBLISHER

```
CREATE TABLE PUBLISHER
(
  Publisher_ID VARCHAR2 (10)  NOT NULL ,
  Publisher_Name VARCHAR2 (50)
)
;

COMMENT ON TABLE PUBLISHER IS 'Publisher'
;

ALTER TABLE PUBLISHER
  ADD CONSTRAINT PUBLISHER_PK PRIMARY KEY ( Publisher_ID) ;
```

g. PURCHASE\_HISTORY

```
CREATE TABLE PURCHASE_HISTORY
(
    Customer_ID VARCHAR2 (6)  NOT NULL ,
    Order_ID VARCHAR2 (6)  NOT NULL
)
;

COMMENT ON TABLE PURCHASE_HISTORY IS 'Purchase History'
;
```

h. SHIPPING\_TYPE

```
CREATE TABLE SHIPPING_TYPE
(
    Shipping_Type VARCHAR2 (10)  NOT NULL ,
    Shipping_Price NUMBER (6)
)
;

COMMENT ON TABLE SHIPPING_TYPE IS 'Shipping Type'
;

ALTER TABLE SHIPPING_TYPE
    ADD CONSTRAINT SHIPPING_TYPE_PK PRIMARY KEY ( Shipping_Type
) ;
```

i. SHOPPING \_CART

```
CREATE TABLE SHOPPING_CART
(
    Shopping_Cart_ID VARCHAR2 (6)  NOT NULL ,
    Book_ID VARCHAR2 (10)  NOT NULL ,
    Price NUMBER (10) ,
    Shopping_cart_Date DATE ,
    Quantity NUMBER (6)
)
;

COMMENT ON TABLE SHOPPING_CART IS 'Shopping Cart'
;

ALTER TABLE SHOPPING_CART
```

```
ADD CONSTRAINT SHOPPING_CART_PK PRIMARY KEY (SHOPPING_CART_ID)
;
```

**2. Fügen Sie den erstellten Tabellen zusätzliche referenzielle Integritäts-Constraints hinzu.**

- a. Nehmen Sie in die Tabelle `BOOKS` ein `FOREIGN KEY`-Constraint auf.

```
ALTER TABLE BOOKS
  ADD CONSTRAINT BOOKS_AUTHOR_FK FOREIGN KEY
  (
    Author_ID
  )
  REFERENCES AUTHOR
  (
    Author_ID
  )
;

ALTER TABLE BOOKS
  ADD CONSTRAINT BOOKS_PUBLISHER_FK FOREIGN KEY
  (
    Publisher_ID
  )
  REFERENCES PUBLISHER
  (
    Publisher_ID
  )
);
```

- b. Nehmen Sie in die Tabelle `ORDER_DETAILS` ein `FOREIGN KEY`-Constraint auf.

```
ALTER TABLE ORDER_DETAILS
  ADD CONSTRAINT Order_ID_FK FOREIGN KEY
  (
    Customer_ID
  )
  REFERENCES CUSTOMER
  (
    Customer_ID
  )
;

ALTER TABLE ORDER_DETAILS
  ADD CONSTRAINT FK_Order_details FOREIGN KEY
```

```

        (
            Shipping_Type
        )
    REFERENCES SHIPPING_TYPE
    (
        Shipping_Type
    )
;

ALTER TABLE ORDER_DETAILS
    ADD CONSTRAINT Order_Details_fk FOREIGN KEY
    (
        Shopping_Cart_ID
    )
    REFERENCES SHOPPING_CART
    (
        Shopping_Cart_ID
    )
;

```

- c. Nehmen Sie in die Tabelle PURCHASE\_HISTORY ein FOREIGN KEY-Constraint auf.

```

ALTER TABLE PURCHASE_HISTORY
    ADD CONSTRAINT Pur_Hist_ORDER_DETAILS_FK FOREIGN KEY
    (
        Order_ID
    )
    REFERENCES ORDER_DETAILS
    (
        Order_ID
    )
;

ALTER TABLE PURCHASE_ HISTORY
    ADD CONSTRAINT Purchase_History_CUSTOMER_FK FOREIGN KEY
    (
        Customer_ID
    )
    REFERENCES CUSTOMER
    (
        Customer_ID
    )
;

```

- d. Nehmen Sie in die Tabelle `SHOPPING_CART` ein FOREIGN KEY-Constraint auf.

```
ALTER TABLE SHOPPING_CART
    ADD CONSTRAINT SHOPPING_CART_BOOKS_FK FOREIGN KEY
    (
        Book_ID
    )
    REFERENCES BOOKS
    (
        Book_ID
    )
;
```

3. Prüfen Sie im Connections Navigator von SQL Developer, ob die Tabellen korrekt erstellt wurden. Blenden Sie im Connections Navigator **Connections** > **myconnection** > **Tables** ein.
4. Erstellen Sie eine Sequence, um die einzelnen Zeilen in der Tabelle `ORDER DETAILS` eindeutig zu kennzeichnen.
- a. Beginnen Sie mit 100, wobei keine Werte im Cache gespeichert werden sollen. Geben Sie der Sequence den Namen `ORDER_ID_SEQ`.

```
CREATE SEQUENCE order_id_seq
START WITH 100
NOCACHE;
```

- b. Überprüfen Sie die Existenz der Sequences im Connections Navigator von SQL Developer.
- Im Connections Navigator sollte der Knoten **myconnection** bereits eingeblendet sein. Blenden Sie **Sequences** ein.

Alternativ können Sie die Data Dictionary View `user_sequences` abfragen:

```
SELECT * FROM user_sequences;
```



5. Geben Sie Daten in die Tabellen ein.

a. Tabelle AUTHOR

Author_ID	Author_Name
AN0001	Oliver Goldsmith
AN0002	Oscar Wilde
AN0003	George Bernard Shaw
AN0004	Leo Tolstoy
AN0005	Percy Shelley
AN0006	Lord Byron
AN0007	John Keats
AN0008	Rudyard Kipling
AN0009	P. G. Wodehouse

AUTHOR_ID	AUTHOR_NAME
1 AN0001	Oliver Goldsmith
2 AN0002	Oscar Wilde
3 AN0003	George Bernard Shaw
4 AN0004	Leo Tolstoy
5 AN0005	Percy Shelley
6 AN0006	Lord Byron
7 AN0007	John Keats
8 AN0008	Rudyard Kipling
9 AN0009	P. G. Wodehouse

b. Tabelle PUBLISHER

Publisher_ID	Publisher_Name
PN0001	Elsevier
PN0002	Penguin Group
PN0003	Pearson Education
PN0004	Cambridge University Press
PN0005	Dorling Kindersley

PUBLISHER_ID	PUBLISHER_NAME
1 PN0001	Elsevier
2 PN0002	Penguin Group
3 PN0003	Pearson Education
4 PN0004	Cambridge University Press
5 PN0005	Dorling Kindersley

c. SHIPPING \_TYPE

Shipping_Type	Shipping_Price
USPS	200
FedEx	250
DHL	150

SHIPPING_TYPE	SHIPPING_PRICE
1 USPS	200
2 FedEx	250
3 DHL	150

d. CUSTOMER

Customer_ID	Customer_Name	Street_Address	City	Phone_number	Credit_Card_Number
CN0001	VelasquezCarmen	283 King Street	Seattle	587-99-6666	000-111-222-333
CN0002	Ngao LaDoris	5 Modrany	Bratislava	586-355-8882	000-111-222-444
CN0003	Nagayama Midori	68 Via Centrale	Sao Paolo	254-852-5764	000-111-222-555
CN0004	Quick-To-See Mark	6921 King Way	Lagos	63-559-777	000-111-222-666
CN0005	Ropeburn Audry	86 Chu Street	Hong Kong	41-559-87	000-111-222-777
CN0006	Urguhart Molly	3035 Laurier Blvd.	Quebec	418-542-9988	000-111-222-888
CN0007	Menchu Roberta	Boulevard de Waterloo 41	Brussels	322-504-2228	000-111-222-999
CN0008	Biri Ben	398 High St.	Columbus	614-455-9863	000-111-222-222
CN0009	Catchpole Antoinette	88 Alfred St.	Brisbane	616-399-1411	000-111-222-111

CUSTOMER_ID	CUSTOMER_NAME	STREET_ADDRESS	CITY	PHONE_NUMBER	CREDIT_CARD_NUMBER
1 CN0001	VelasquezCarmen	283 King Street	Seattle	587-99-6666	000-111-222-333
2 CN0002	Ngao LaDoris	5 Modrany	Bratislava	586-355-8882	000-111-222-444
3 CN0003	Nagayama Midori	68 Via Centrale	Sao Paolo	254-852-5764	000-111-222-555
4 CN0004	Quick-To-See Mark	6921 King Way	Lagos	63-559-777	000-111-222-666
5 CN0005	Ropeburn Audry	86 Chu Street	Hong Kong	41-559-87	000-111-222-777
6 CN0006	Urguhart Molly	3035 Laurier Blvd.	Quebec	418-542-9988	000-111-222-888
7 CN0007	Menchu Roberta	Boulevard de Waterloo 41	Brussels	322-504-2228	000-111-222-999
8 CN0008	Biri Ben	398 High St.	Columbus	614-455-9863	000-111-222-222
9 CN0009	Catchpole Antoinette	88 Alfred St.	Brisbane	616-399-1411	000-111-222-111

e. CREDIT\_CARD\_DETAILS

Credit_Card_Number	Credit_Card_Type	Expiry_Date
000-111-222-333	VISA	17-JUN-2009
000-111-222-444	MasterCard	24-SEP-2005
000-111-222-555	AMEX	11-JUL-2006

000-111-222-666	VISA	22-OCT-2008
000-111-222-777	AMEX	26-AUG-2000
000-111-222-888	MasterCard	15-MAR-2008
000-111-222-999	VISA	4-AUG-2009
000-111-222-111	Maestro	27-SEP-2001
000-111-222-222	AMEX	9-AUG-2004

R2	CREDIT_CARD_NUMBER	R2	CREDIT_CARD_TYPE	R2	EXPIRY_DATE
1	000-111-222-333		VISA		17-JUN-09
2	000-111-222-444		MasterCard		24-SEP-05
3	000-111-222-555		AMEX		11-JUL-06
4	000-111-222-666		VISA		22-OCT-08
5	000-111-222-777		AMEX		26-AUG-00
6	000-111-222-888		MasterCard		15-MAR-08
7	000-111-222-999		VISA		04-AUG-09
8	000-111-222-111		Maestro		27-SEP-01
9	000-111-222-222		AMEX		09-AUG-04

f. BOOKS

Book_ID	Book_Name	Author_ID	Price	Publisher_ID
BN0001	Florentine Tragedy	AN0002	150	PN0002
BN0002	A Vision	AN0002	100	PN0003
BN0003	Citizen of the World	AN0001	100	PN0001
BN0004	The Complete Poetical Works of Oliver Goldsmith	AN0001	300	PN0001
BN0005	Androcles and the Lion	AN0003	90	PN0004
BN0006	An Unsocial Socialist	AN0003	80	PN0004
BN0007	A Thing of Beauty is a Joy Forever	AN0007	100	PN0002
BN0008	Beyond the Pale	AN0008	75	PN0005
BN0009	The Clicking of Cuthbert	AN0009	175	PN0005
BN00010	Bride of Frankenstein	AN0006	200	PN0001

BN00011	Shelley's Poetry and Prose	AN0005	150	PN0003
BN00012	War and Peace	AN0004	150	PN0002

	BOOK_ID	BOOK_NAME	AUTHOR_ID	PRICE	PUBLISHER_ID
1	BN0001	Florentine Tragedy	AN0002	150	PN0002
2	BN0002	A Vision	AN0002	100	PN0003
3	BN0003	Citizen of the World	AN0001	100	PN0001
4	BN0004	The Complete Poetical Works of Oliver Goldsmith	AN0001	300	PN0001
5	BN0005	Androcles and the Lion	AN0003	90	PN0004
6	BN0006	An Unsocial Socialist	AN0003	80	PN0004
7	BN0007	A Thing of Beauty is a Joy Forever	AN0007	100	PN0002
8	BN0008	Beyond the Pale	AN0008	75	PN0005
9	BN0009	The Clicking of Cuthbert	AN0009	175	PN0005
10	BN0010	Bride of Frankenstein	AN0006	200	PN0001
11	BN0011	Shelley Poetry and Prose	AN0005	150	PN0003
12	BN0012	War and Peace	AN0004	150	PN0002

g. SHOPPING\_CART

Shopping_Cart_ID	Book_ID	Price	Shopping_Cart_Date	Quantity
SC0001	BN0002	200	12-JUN-2001	10
SC0002	BN0003	90	31-JUL-2004	8
SC0003	BN0003	175	28-JUN-2005	7
SC0004	BN0001	80	14-AUG-2006	9
SC0005	BN0001	175	21-SEP-2006	4
SC0006	BN0004	100	11-AUG-2007	6
SC0007	BN0005	200	28-OCT-2007	5
SC0008	BN0006	100	25-NOV-2009	7
SC0009	BN0006	150	18-SEP-2009	8

	SHOPPING_CART_ID	BOOK_ID	PRICE	SHOPPING_CART_DATE	QUANTITY
1	SC0001	BN0002	200	12-JUN-01	10
2	SC0002	BN0003	90	31-JUL-05	8
3	SC0003	BN0003	175	28-JUN-05	7
4	SC0004	BN0001	80	14-AUG-06	9
5	SC0005	BN0001	175	21-SEP-06	4
6	SC0006	BN0004	100	11-AUG-07	6
7	SC0007	BN0005	200	28-OCT-07	5
8	SC0008	BN0006	100	25-NOV-09	7
9	SC0009	BN0006	150	18-SEP-09	8

h. ORDER \_DETAILS

Order_ID	Customer_ID	Shipping_ Typ	Date _of _Purchase	Shopping _Cart_ID
OD0001	CN0001	USPS	12-JUN-2001	SC0002
OD0002	CN0002	USPS	28-JUN-2005	SC0005
OD0003	CN0003	FedEx	31-JUL-2004	SC0007
OD0004	CN0004	FedEx	14-AUG-2006	SC0004
OD0005	CN0005	FedEx	21-SEP-2006	SC0003
OD0006	CN0006	DHL	28-OCT-2007	SC0001
OD0007	CN0007	DHL	11-AUG-2007	SC0006
OD0008	CN0008	DHL	18-SEP-2009	SC0008
OD0009	CN0009	USPS	25-NOV-2009	SC0009

	ORDER_ID	CUSTOMER_ID	SHIPPING_TYPE	DATE_OF_PURCHASE	SHOPPING_CART_ID
1	OD0001	CN0001	USPS	12-JUN-01	SC0002
2	OD0002	CN0002	USPS	28-JUN-05	SC0005
3	OD0003	CN0003	FedEx	31-JUL-05	SC0007
4	OD0004	CN0004	FedEx	14-AUG-06	SC0004
5	OD0005	CN0005	FedEx	21-SEP-06	SC0003
6	OD0006	CN0006	DHL	28-OCT-07	SC0001
7	OD0007	CN0007	DHL	11-AUG-07	SC0006
8	OD0008	CN0008	DHL	18-SEP-09	SC0008
9	OD0009	CN0009	USPS	25-NOV-09	SC0009

i. PURCHASE\_HISTORY

Customer_ID	Order_ID
CN0001	OD0001
CN0003	OD0002
CN0004	OD0005
CN0009	OD0007

	CUSTOMER_ID	ORDER_ID
1	CN0001	OD0001
2	CN0003	OD0002
3	CN0004	OD0005
4	CN0009	OD0007

6. Erstellen Sie die View `CUSTOMER_DETAILS`, in der Name und Adresse der Kunden sowie detaillierte Informationen zur Bestellung des jeweiligen Kunden angezeigt werden. Sortieren Sie die Ergebnisse nach der Kunden-ID.

```
CREATE VIEW customer_details AS
    SELECT c.customer_name, c.street_address, o.order_id,
    o.customer_id, o.shipping_type, o.date_of_purchase,
    o.shopping_cart_id
    FROM customer c JOIN order_details o
    ON c.customer_id = o.customer_id;

SELECT *
FROM customer_details
ORDER BY customer_id;
```

	CUSTOMER_NAME	STREET_ADDRESS	ORDER_ID	CUSTOMER_ID	SHIPPING_TYPE	DATE_OF_PURCHASE	SHOPPING_CART_ID
1	VelasquezCarmen	283 King Street	0D0001	CN0001	USPS	12-JUN-01	SC0002
2	Ngao LaDoris	5 Modrany	0D0002	CN0002	USPS	28-JUN-05	SC0005
3	Nagayama Midori	68 Via Centrale	0D0003	CN0003	FedEx	31-JUL-05	SC0007
4	Quick-To-See Mark	6921 King Way	0D0004	CN0004	FedEx	14-AUG-06	SC0004
5	Ropeburn Audry	86 Chu Street	0D0005	CN0005	FedEx	21-SEP-06	SC0003
6	Urguhart Molly	3035 Laurier Blvd.	0D0006	CN0006	DHL	28-OCT-07	SC0001
7	Menchu Roberta	Boulevard de Waterloo 41	0D0007	CN0007	DHL	11-AUG-07	SC0006
8	Biri Ben	398 High St.	0D0008	CN0008	DHL	18-SEP-09	SC0008
9	Catchpole Antoinette	88 Alfred St.	0D0009	CN0009	USPS	25-NOV-09	SC0009

7. Ändern Sie Daten in den Tabellen.

- a. Fügen Sie Details zu einem neuen Buch hinzu. Prüfen Sie, ob die Angabe zum Autor des Buchs in der Tabelle `AUTHOR` verfügbar ist. Erstellen Sie gegebenenfalls einen Eintrag in der Tabelle `AUTHOR`.

```
INSERT INTO books(book_id, book_name, author_id, price,
publisher_id)
VALUES ('BN0013', 'Two States', 'AN0009', '150', 'PN0005');
```

	BOOK_ID	BOOK_NAME	AUTHOR_ID	PRICE	PUBLISHER_ID
1	BN0001	Florentine Tragedy	AN0002	150	PN0002
2	BN0002	A Vision	AN0002	100	PN0003
3	BN0003	Citizen of the World	AN0001	100	PN0001
4	BN0004	The Complete Poetical Works of Oliver Goldsmith	AN0001	300	PN0001
5	BN0005	Androcles and the Lion	AN0003	90	PN0004
6	BN0006	An Unsocial Socialist	AN0003	80	PN0004
7	BN0007	A Thing of Beauty is a Joy Forever	AN0007	100	PN0002
8	BN0008	Beyond the Pale	AN0008	75	PN0005
9	BN0009	The Clicking of Cuthbert	AN0009	175	PN0005
10	BN0010	Bride of Frankenstein	AN0006	200	PN0001
11	BN0011	Shelley Poetry and Prose	AN0005	150	PN0003
12	BN0012	War and Peace	AN0004	150	PN0002
13	BN0013	Two States	AN0009	150	PN0005

- b. Geben Sie Warenkorbdetails für das Buch ein, das Sie gerade im Schritt 7(a) angelegt haben.

```
INSERT INTO shopping_cart(shopping_cart_id, book_id, price,
Shopping_cart_date,quantity)
VALUES ('SC0010', 'BN0013', '200', TO_DATE('12-JUN-2006', 'DD-MON-
YYYY'), '12');
```

8. Erstellen Sie einen Bericht, der die Kaufhistorie der einzelnen Kunden enthält. Nehmen Sie Kundenname und -ID, Buch-ID, Kaufdatum und Warenkorb-ID in den Bericht auf. Speichern Sie die Befehle für die Generierung des Berichts in der Skriptdatei `lab_apcs_8.sql`.

**Hinweis:** Ihre Ergebnisse können abweichen.

```
SELECT  c.customer_name CUSTOMER, c.customer_id,
s.shopping_cart_id, s.book_id,o.date_of_purchase
FROM    customer c
JOIN    order_details o
ON      o.customer_id=c.customer_id
JOIN    shopping_cart s
ON      o.shopping_cart_id=s.shopping_cart_id;
```

