Programmierung von CAx-Systemen

SQL-Datenbanken mit C++/Qt

HM

1. Einleitung
2. Messwerte-Datenbank
3. Structured Query Language, SQL
4. SQL mit C++/Qt
5. Übungen

HM

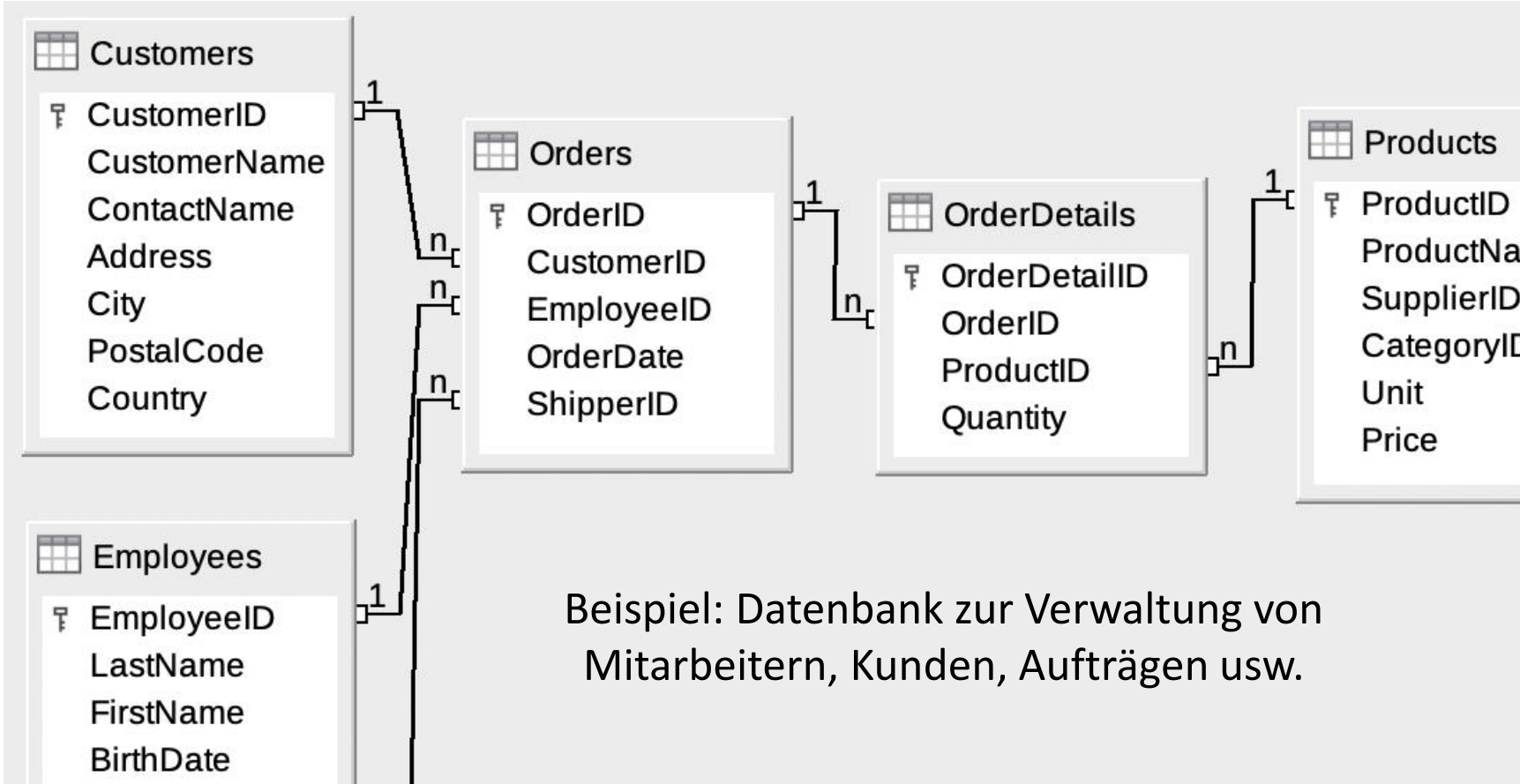
Relationale Datenbanken

A relational database is a (most commonly digital) database based on the relational model of data, as proposed by E. F. Codd in 1970. (...) Many relational database systems are equipped with the option of using SQL (Structured Query Language) for querying and updating the database.

Quelle: [1]

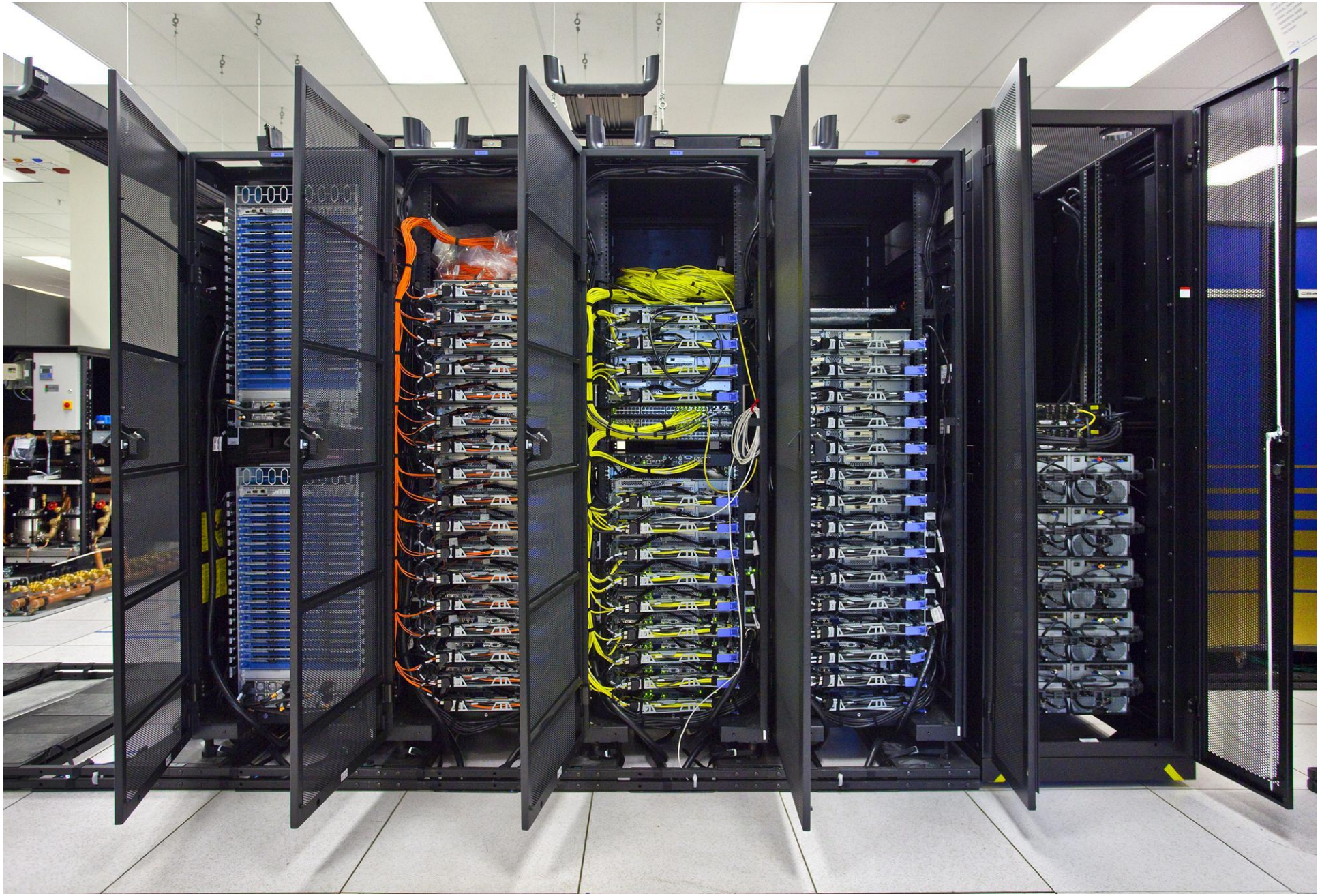
SQL (...) ist eine Datenbank-sprache zur Definition von Datenstrukturen in relatio-nalen Datenbanken sowie zum Bearbeiten (Einfügen, Verändern, Löschen) und Abfragen von darauf basie-renden Datenbeständen.

Quelle: [2]



Beispiel: Datenbank zur Verwaltung von Mitarbeitern, Kunden, Aufträgen usw.

Quelle: [3]



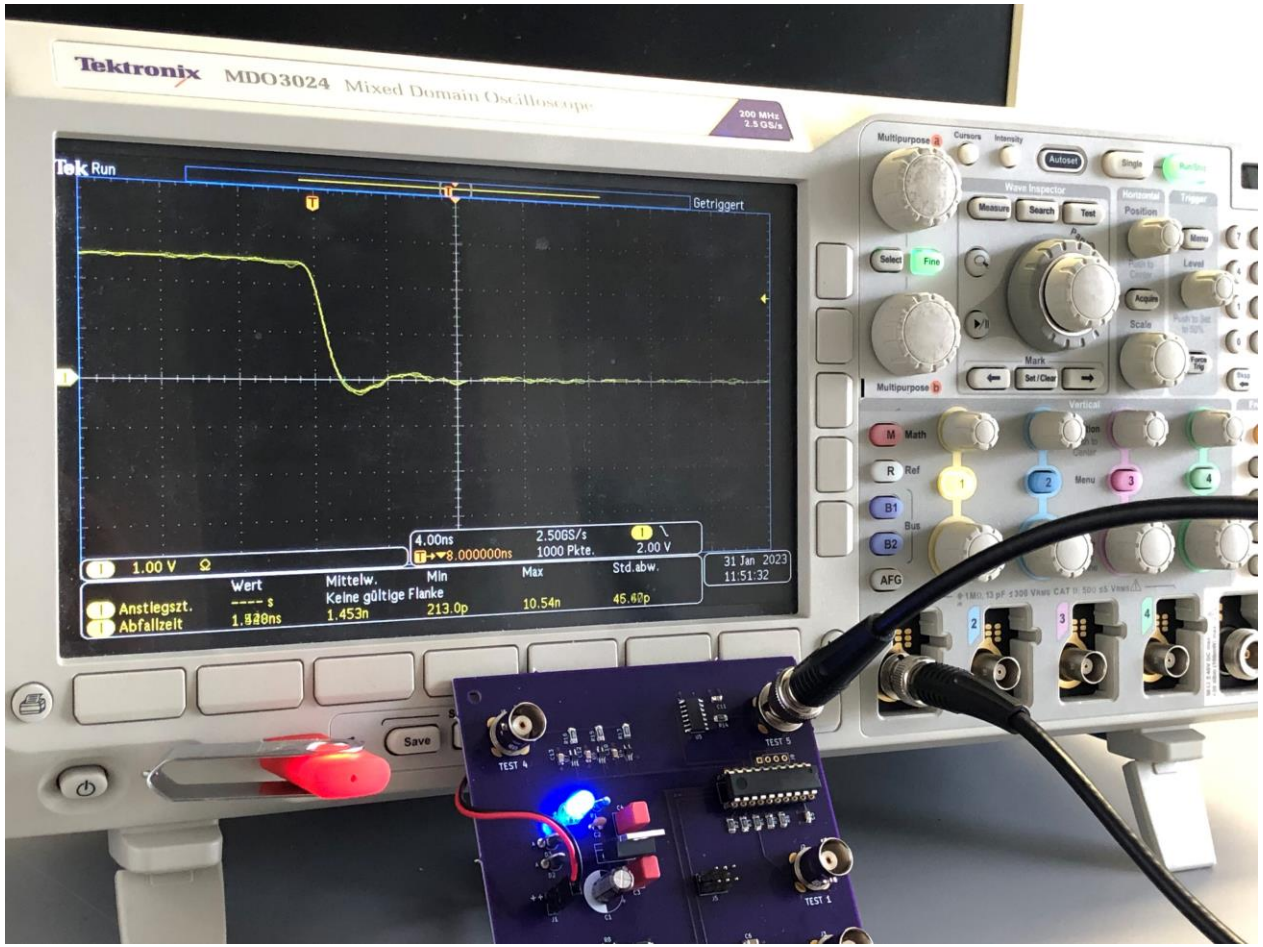
Quelle: [4]

1. Einleitung
2. Messwerte-Datenbank
3. Structured Query Language, SQL
4. SQL mit C++/Qt
5. Übungen

HM

Datenbank für Oszilloskop-Messwerte

Die mit einem Oszilloskop aufgenommenen Spannungsmesswerte sollen in einer Datenbank gespeichert werden. So können sie leicht ausgewertet und geeignet visualisiert werden.



tek0000CH1.csv

Datei Bearbeiten Ansicht

```
Model,MDO3024
Firmware Version,1.20

Waveform Type,ANALOG
Point Format,Y
Horizontal Units,s
Horizontal Scale,4e-09
Horizontal Delay,8e-09
Sample Interval,4e-10
Record Length,1000
Gating,0.1% to 100.0%
Probe Attenuation,1
Vertical Units,V
Vertical Offset,0
Vertical Scale,2
Vertical Position,0
,
,
,
Label,
TIME,CH1
-1.920000e-07,0.08
-1.916000e-07,0
-1.912000e-07,0
-1.908000e-07,0
-1.904000e-07,0
-1.900000e-07,0
-1.896000e-07,0
-1.892000e-07,0.08
-1.888000e-07,0
-1.884000e-07,0
```

Variante A, nur eine Datenbanktabelle, besonders einfach …

Es ist grundsätzlich möglich, alle Daten in einer einzigen Tabelle abzulegen.
Was sind die Vor- und Nachteile dieser Variante?

Tabelle MEASUREMENT_DATA

RECORD_NO	X_POSITION	VOLTAGE	MODEL_FIRMWARE	CHANNEL	Y_OFFSET	Y_ATTN
1	-1,9200E-07	0,08	MDO3024 1.20	CH1	0,00	1,00
1	-1,9160E-07	0,00	MDO3024 1.20	CH1	0,00	1,00
1	-1,9120E-07	0,00	MDO3024 1.20	CH1	0,00	1,00
1	-1,9080E-07	0,00	MDO3024 1.20	CH1	0,00	1,00
1	-1,9040E-07	0,00	MDO3024 1.20	CH1	0,00	1,00
1	-1,9000E-07	0,00	MDO3024 1.20	CH1	0,00	1,00
1	-1,8960E-07	0,00	MDO3024 1.20	CH1	0,00	1,00
1	-1,8920E-07	0,08	MDO3024 1.20	CH1	0,00	1,00
1	-1,8880E-07	0,00	MDO3024 1.20	CH1	0,00	1,00

Variante B, erste Normalform

Quelle: [5]

1. Jede Spalte enthält nur unteilbare (atomare, atomische) Werte.
2. Spalten, die gleiche oder gleichartige Informationen enthalten, sind in eigene Tabellen (Relationen) auszulagern.
3. Jede Tabelle enthält einen Primärschlüssel.

= Primärschlüssel

= Fremdschlüssel

Tabelle MEASUREMENT_DATA

ID	RECORD_NO	X_POSITION	VOLTAGE	DEVICE_ID	CHANNEL_ID
1	1	-1,9200E-07	0,08	1	1
2	1	-1,9160E-07	0	1	1
3	1	-1,9120E-07	0	1	1
4	1	-1,9080E-07	0	1	1
5	1	-1,9040E-07	0	1	1
6	1	-1,9000E-07	0	1	1
7	1	-1,8960E-07	0	1	1
8	1	-1,8920E-07	0,08	1	1
9	1	-1,8880E-07	0	1	1

Tabelle DEVICE

ID	MODEL	FIRMWARE
1	MDO3024	1.20

Tabelle CHANNEL

ID	NAME	OFFSET_V	ATTN_V
1	CH1	0,00	1,00
2	CH2	0,00	1,00

Zweite Normalform

Quelle: [5]

1. Die Tabelle erfüllt die 1. Normalform.
2. Alle Informationen in den Spalten, die nicht Teil des Primärschlüssels sind, müssen sich auf den gesamten Primärschlüssel beziehen.

Die zweite Normalform kann ganz einfach dadurch gewährleistet werden, dass sich der Primärschlüssel nur auf eine Spalte bezieht.

Dritte Normalform

Quelle: [5]

1. Die Tabelle erfüllt die 2. Normalform.
2. Informationen in den Spalten, die nicht Teil des Primärschlüssels sind, dürfen funktional nicht voneinander abhängen.

i Es gibt eine Reihe von „reservierten“ Bezeichnern, die nicht als Namen von Tabellen oder Spalten verwendet werden sollten: https://en.wikipedia.org/w/index.php?title=List_of_SQL_reserved_words&oldid=1135243723 (Abgerufen: 17. Mai 2023)

1. Einleitung
2. Messwerte-Datenbank
3. Structured Query Language, SQL
4. SQL mit C++/Qt
5. Übungen

HM

Tabellen anlegen und löschen (a)

```
CREATE TABLE Device (
        Id INTEGER PRIMARY KEY,
        Model VARCHAR(50),
        Firmware FLOAT );

DROP TABLE Device;
```

Tabelle wieder entfernen

Tabelle DEVICE

ID	MODEL	FIRMWARE
1	MDO3024	1.20

Tabelle CHANNEL

ID	NAME	OFFSET_V	ATTN_V
1	CH1	0,00	1,00
2	CH2	0,00	1,00

i Durch (unbeabsichtigtes) Überschreiben oder Löschen von Tabellen kann es zu Datenverlust kommen.

Die wichtigsten SQL-Datentypen

INTEGER	ganze Zahlen
VARCHAR(n)	Zeichenkette, max. n Zeichen
FLOAT	Fließkommazahl

Aufgabe

Wie lautet die SQL-Anweisung zum Anlegen der CHANNEL-Tabelle?

Tabellen anlegen und löschen (b)

Die Datensätze in der Tabelle MEASUREMENT_DATA verweisen über Fremdschlüssel auf bestimmte Einträge in den Tabellen DEVICE und CHANNEL. Diese Fremdschlüssel werden bei der Definition der Tabelle MEASUREMENT_DATA angegeben.

```
CREATE TABLE MeasurementData (
        Id INTEGER PRIMARY KEY,
        RecordNo INTEGER,
        XPosition FLOAT,
        Voltage FLOAT,
        DeviceId INTEGER,
        ChannelId INTEGER,
        FOREIGN KEY (DeviceId) REFERENCES Device(Id),
        FOREIGN KEY (ChannelId) REFERENCES Channel(Id) );
```

Tabelle MEASUREMENT_DATA

ID	RECORD_NO	X_POSITION	VOLTAGE	DEVICE_ID	CHANNEL_ID
1	1	-1,9200E-07	0,08	1	1
2	1	-1,9160E-07	0	1	1
3	1	-1,9120E-07	0	1	1

```
.schema MeasurementData --indent
```

← *Tabellenstruktur anzeigen*

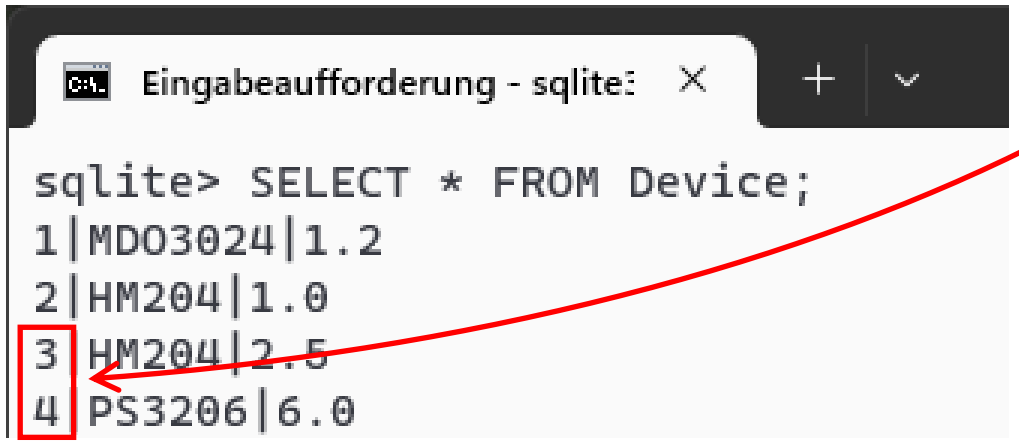
Daten einfügen und abfragen (a)

- INSERT INTO Device (Id, Model, Firmware) VALUES (1, 'MDO3024', 1.2);
- INSERT INTO Device (Id, Model, Firmware) VALUES (2, 'HM204', 1.0);

Die Tabelle Device besitzt einen INTEGER-Primärschlüssel (Id). Wenn die Id bei der INSERT-Anweisung nicht angegeben wird, dann vergibt SQLite automatisch eindeutige Werte.

- INSERT INTO Device (Model, Firmware) VALUES ('HM204', 2.5);
- INSERT INTO Device (Model, Firmware) VALUES ('PS3206', 6.0);

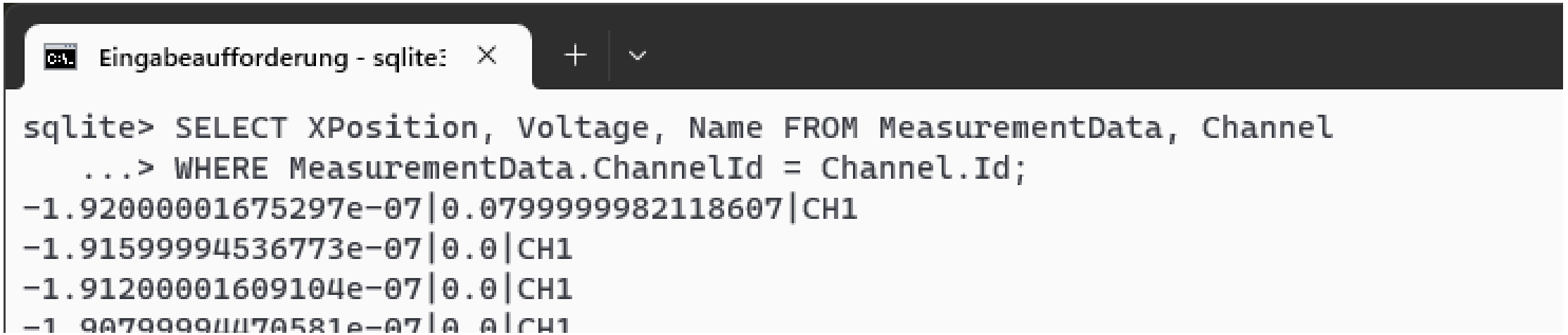
Mittels SELECT können die eingegebenen Daten wieder abgefragt werden.



Daten einfügen und abfragen (b)

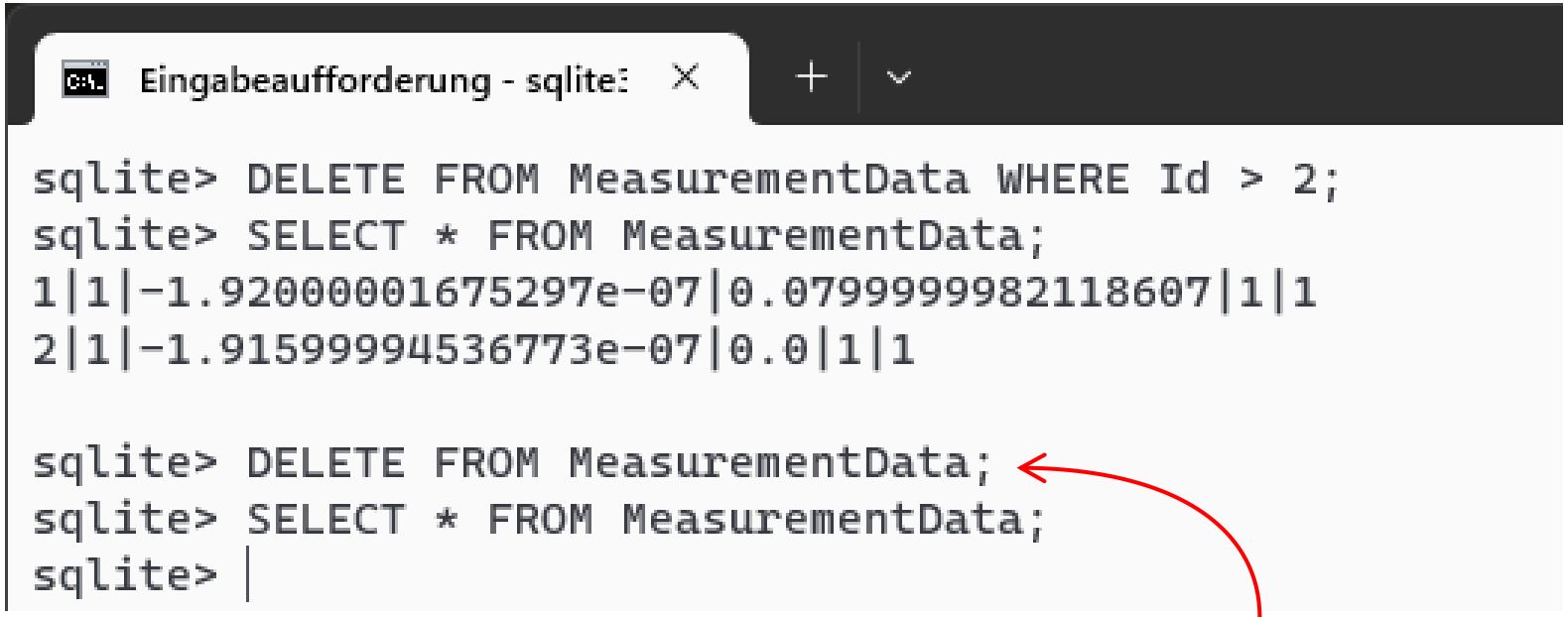
Bei Abfragen mittels SELECT können Bedingungen angegeben werden.

Es ist möglich, bei einer Abfrage mehrere Tabellen zu verknüpfen.



Daten aus Tabellen löschen

Mit der Anweisung DELETE FROM werden Einträge aus Datenbanktabellen gelöscht.



Vergessen Sie nicht, mittels WHERE anzugeben, welche Einträge gelöscht werden sollen. Fehlt diese Bedingung, so werden aus der angegebenen Tabelle alle Einträge gelöscht.

1. Einleitung
2. Messwerte-Datenbank
3. Structured Query Language, SQL
4. SQL mit C++/Qt
5. Übungen

HM

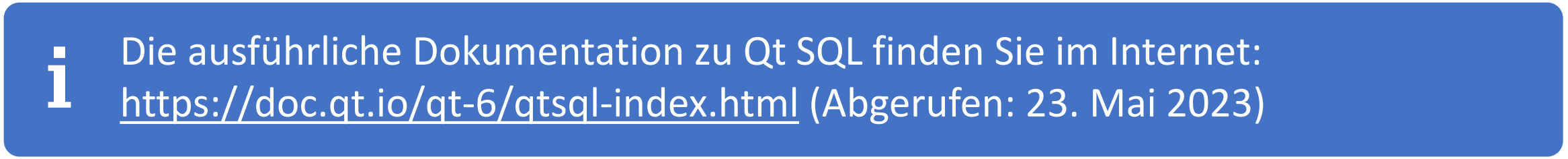
Datensätze schreiben und lesen mit Qt SQL

Das folgende Programm erzeugt eine SQLite-Datenbank, schreibt einige Datensätze und liest sie anschließend wieder aus. Versuchen Sie, den Quelltext zu verstehen.

```
#include <iostream>
#include <stdlib.h>  // EXIT_FAILURE
using namespace std;

#include <QSqlDatabase>
#include <QSqlQuery>
#include <QSqlError>
#include <QVariant>

int main()
{
	// Datenbank-Verbindung öffnen, ggf. neue SQLite-Datei anlegen
	QSqlDatabase db = QSqlDatabase::addDatabase("QSQLITE");
	db.setDatabaseName("bsp1.db");
	if(db.open() == false)
	{
		cout << "Unable to open database" << endl;
		exit(EXIT_FAILURE);
	}
```

```
// Tabelle anlegen, ggf. Fehlermeldung ausgeben
bool ok;
QSqlQuery query(db);
ok = query.exec("CREATE TABLE Device (Id INTEGER PRIMARY KEY,
                "Model VARCHAR(50), Firmware FLOAT )");
if(!ok)
{
       cout << query.lastError().text().toStdString() << endl;
}

// Datensätze in Tabelle schreiben
query.prepare("INSERT INTO Device (Model, Firmware) "
              "VALUES (:Model, :Firmware)");

query.bindValue(":Model", "MDO3024");
query.bindValue(":Firmware", 1.2);
query.exec();

query.bindValue(":Model", "MDO3024");
query.bindValue(":Firmware", 2.0);
query.exec();
```

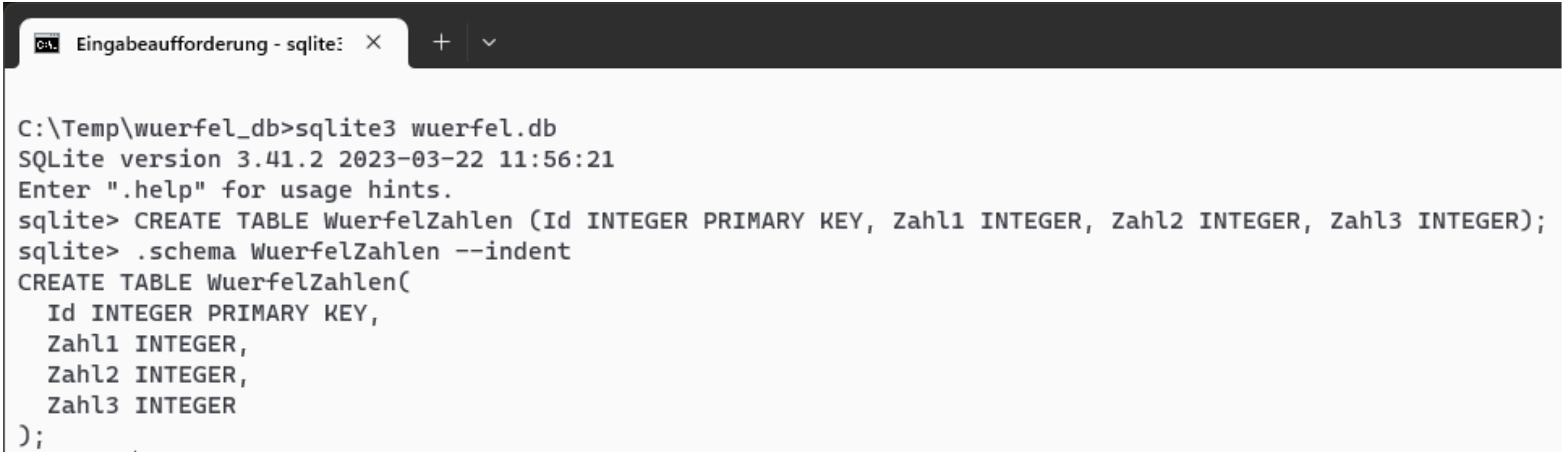
```
        // Datensätze aus Tabelle auslesen
        query.exec("SELECT Id, Model, Firmware FROM Device");
        while(query.next())
        {
                int    id       = query.value(0).toInt();
                string model    = query.value(1).toString().toStdString();
                double firmware = query.value(2).toDouble();

                cout << id << ", " << model << ", " << firmware << endl;
        }

        // Verbindung zur Datenbank schließen
        db.close();
}
```



1. Einleitung
2. Messwerte-Datenbank
3. Structured Query Language, SQL
4. SQL mit C++/Qt
5. Übungen

Übung 1, Würfel-Statistik (a)

Erzeugen Sie mittels sqlite3.exe eine neue Datenbank zum Abspeichern von Zufallszahlen. Jeder Datensatz besteht aus drei INTEGER-Zahlen.



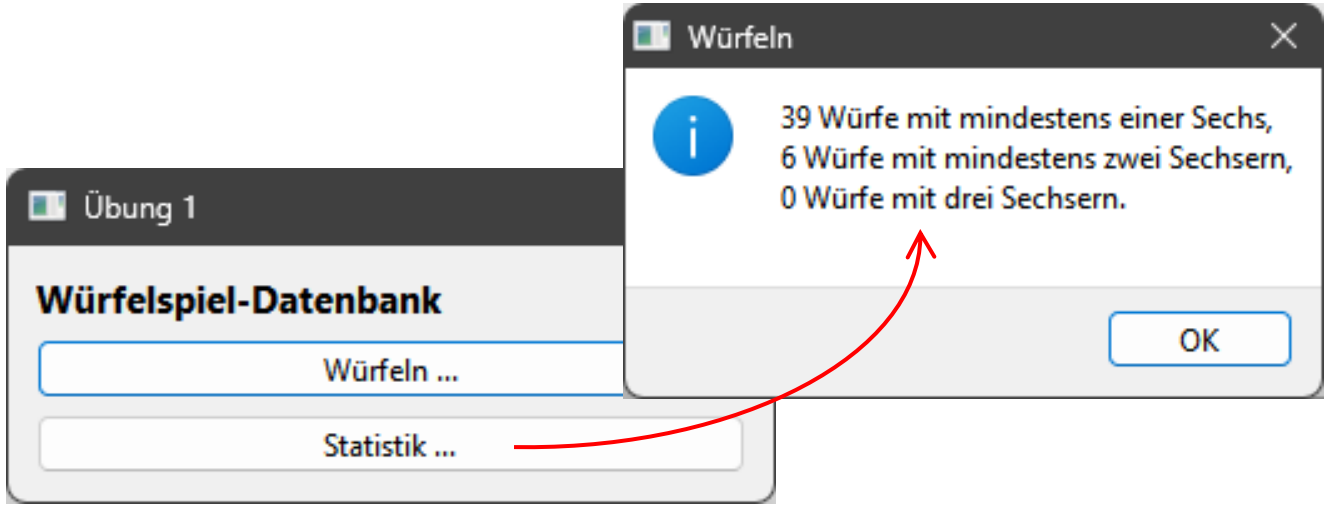
Übung 1, Würfel-Statistik (b)

Erstellen Sie eine Qt-Dialog-Applikation mit zwei Schaltflächen „Würfeln“ und „Statistik“.

i. Beim Programmstart öffnet sich ein „QFileDialog“ zur Auswahl einer Datenbankdatei. Wenn keine Datei ausgewählt wird oder die Datei nicht geöffnet werden kann, erfolgt eine Fehlermeldung und das Programm wird beendet.

ii. Nach Druck auf die Schaltfläche „Würfeln“ werden zunächst alle vorhandenen Datensätze gelöscht (das Ergebnis ist also eine leere Tabelle). Nun werden 100 Datensätze mit zufälligen INTEGER-Werten in der Datenbank abgespeichert (alle Zahlen im Bereich von 1 … 6, wie bei einem Würfelspiel).

iii. Nach Druck auf die Schaltfläche „Statistik“ wird die Anzahl der Datensätze mit „Sechsern“ auf dem Bildschirm ausgegeben.



Übung 1, Würfel-Statistik (c) – Tipps zu Punkt i.

```
// Datei main.cpp
#include "dialog.h"
#include <QApplication>

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    Dialog w;
    if(w.db.isOpen() == true)
    {
        w.show();
        a.exec();
    }
    return 0;
}
```

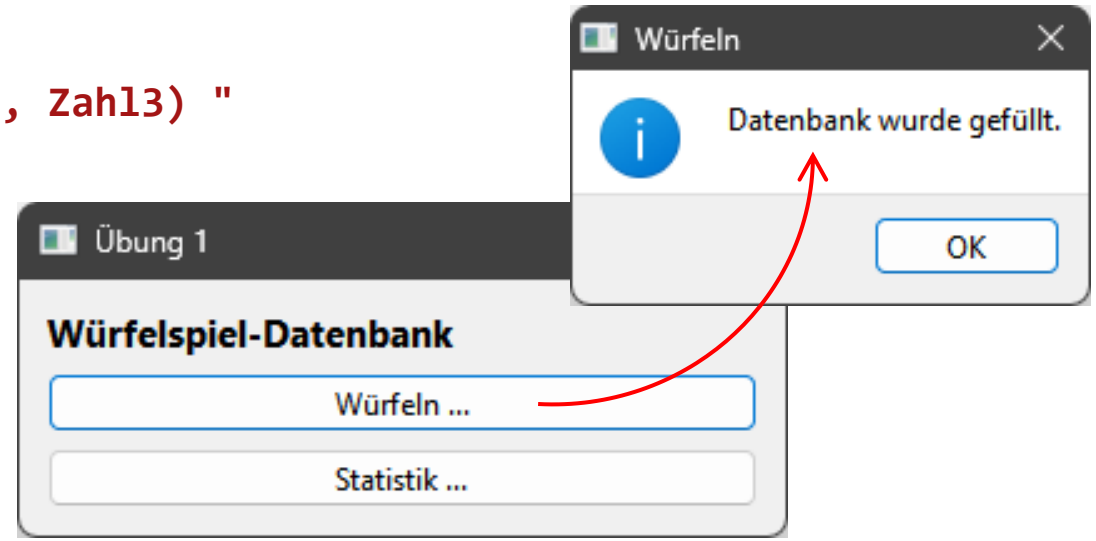
```
# Datei U1-Wuerfel.pro
QT      += core gui
QT      += sql

CONFIG += c++17
```

```
// Datei dialog.h
#ifndef DIALOG_H
#define DIALOG_H

#include <QDialog>
#include <QSqlDatabase>
#include <QSqlQuery>
#include <QFileDialog>
#include <QMessageBox>
#include <stdlib.h>
#include <sstream>

QT_BEGIN_NAMESPACE
namespace Ui { class Dialog; }
QT_END_NAMESPACE

class Dialog : public QDialog
{
    Q_OBJECT

    public:
    QSqlDatabase db;
```

```
// Datei dialog.cpp
#include "dialog.h"
#include "ui_dialog.h"

Dialog::Dialog(QWidget *parent)
: QDialog(parent), ui(new Ui::Dialog)
{
    ui->setupUi(this);

    QString dbFile =
      QFileDialog::getOpenFileName(
        this, "Datenbank");

    if(dbFile.size() > 0) {
        db = QSqlDatabase::
          addDatabase("QSQLITE");
        db.setDatabaseName(dbFile);
        db.open();
    }

    if(db.isOpen() == false) {
        QMessageBox::warning(
          this, "Datenbank",
          "Öffnen fehlgeschlagen.");
    }
}
```

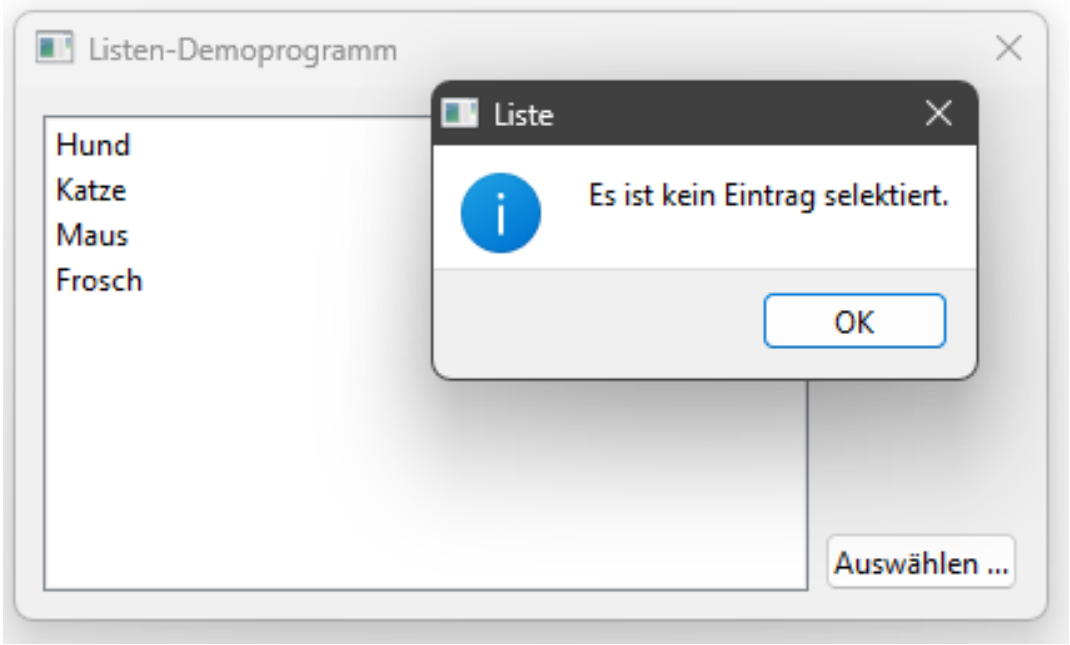
Übung 1, Würfel-Statistik (d) – Tipps zu Punkt ii.

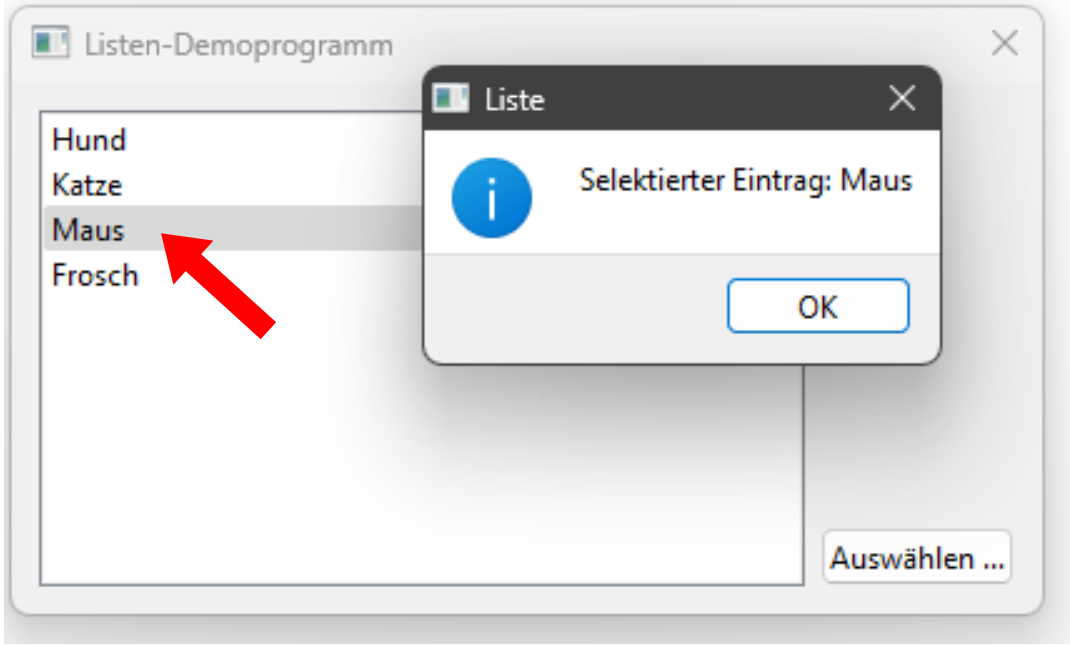
```
// Datei dialog.cpp
void Dialog::on_b_wuerfeln_clicked()
{
    QSqlQuery query(db);
    query.exec("DELETE FROM WuerfelZahlen");

    query.prepare("INSERT INTO WuerfelZahlen (Zahl1, Zahl2, Zahl3) "
                  "VALUES (:Zahl1, :Zahl2, :Zahl3)");

    for(int i = 0; i < 100; ++i)
    {
        query.bindValue(":Zahl1", rand() % 6 + 1);
        query.bindValue(":Zahl2", rand() % 6 + 1);
        query.bindValue(":Zahl3", rand() % 6 + 1);
        query.exec();
    }

    QMessageBox::information(this, "Würfeln", "Datenbank wurde gefüllt.");
}
```



Übung 2, QListWidget

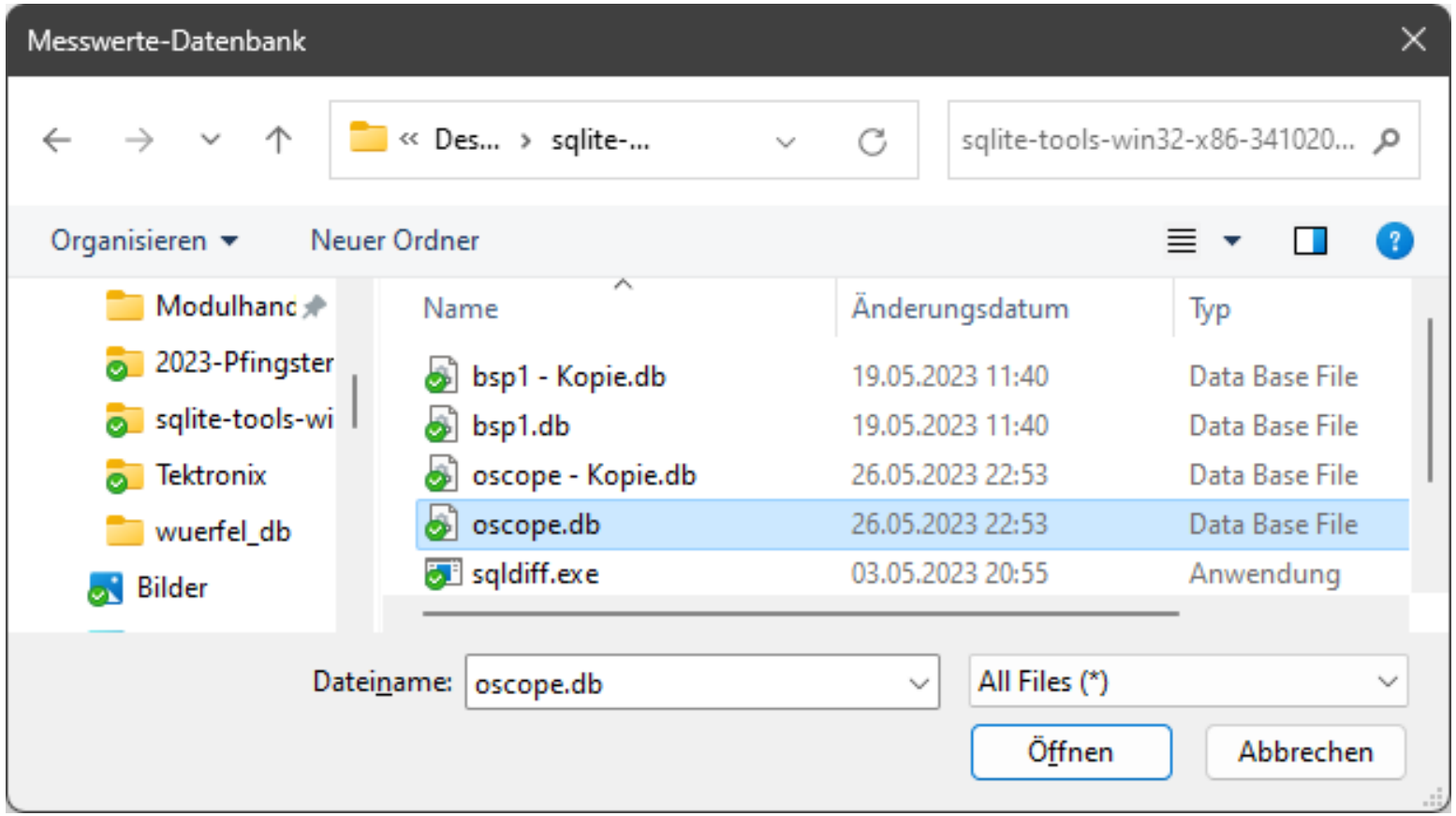
Erstellen Sie eine neue Dialog-Applikation und versuchen Sie zu verstehen, wie ein Eintrag in einem QListWidget ausgewählt werden kann. In der Übung 3 wird dies benötigt …





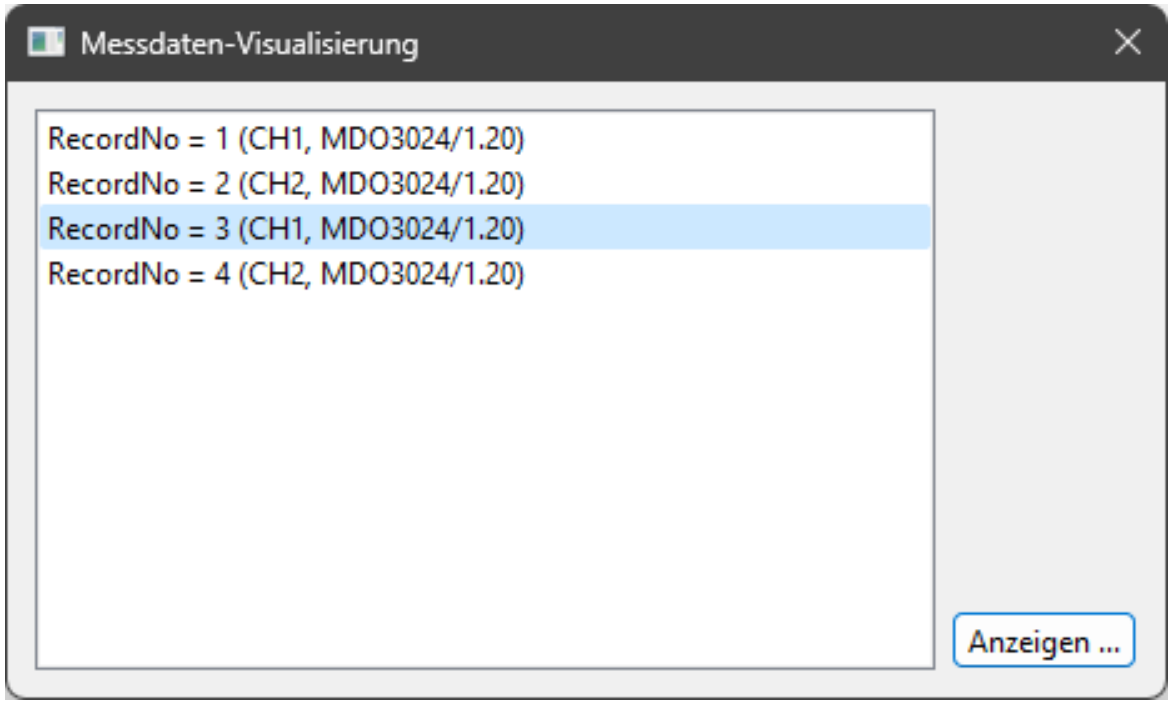
Übung 3, Messwerte-Visualisierung (a)

Erstellen Sie eine Applikation zur Visualisierung von Oszilloskop-Messungen, die in einer Messwerte-Datenbank gespeichert sind. Direkt nach dem Start des Programms öffnet sich ein „QFileDialog“ zur Auswahl der Datenbankdatei:



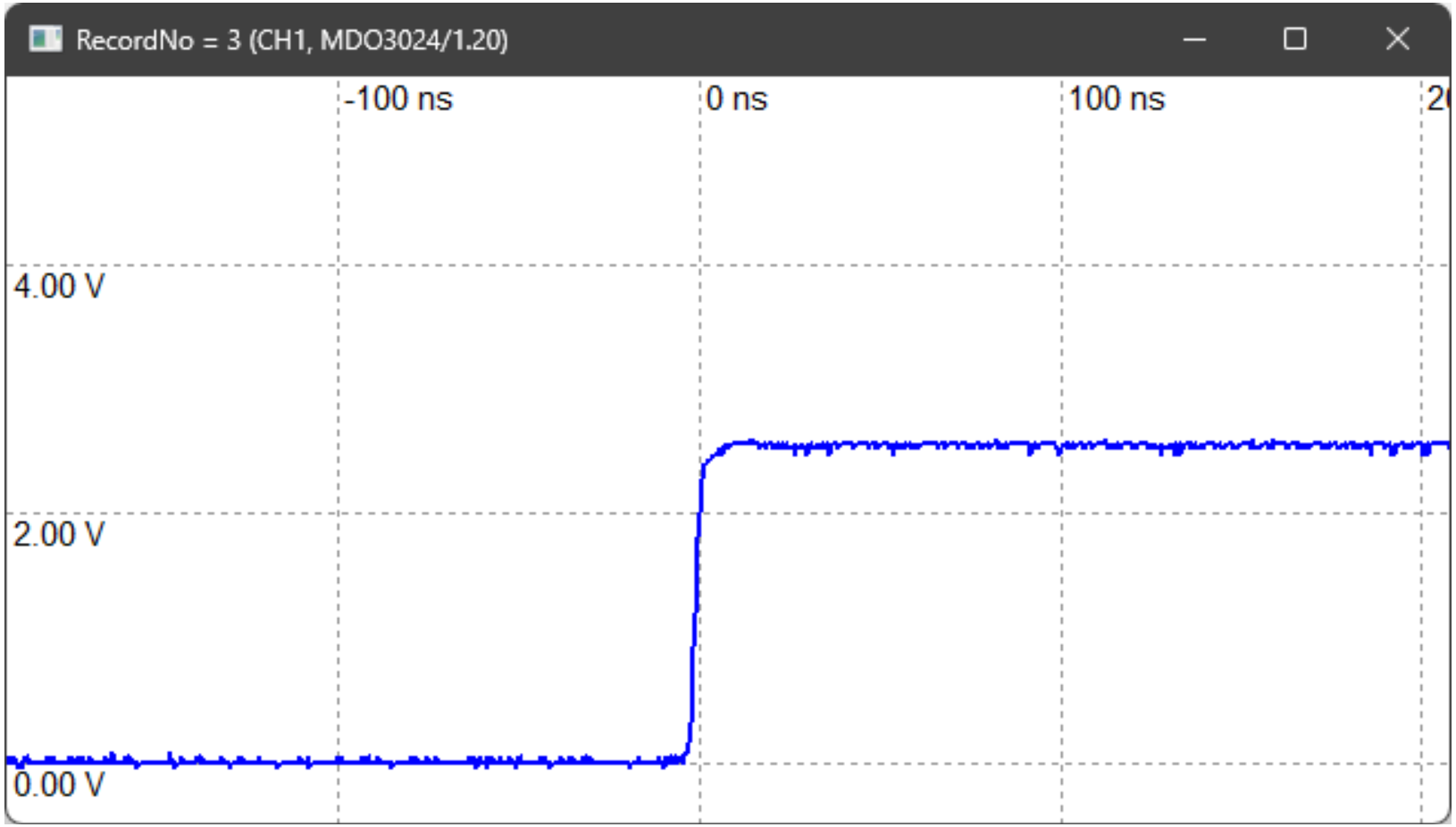
Übung 3, Messwerte-Visualisierung (b)

Im Dialogfenster werden alle Oszilloskop-Messungen aufgelistet, die in der Messwerte-Datenbank gespeichert sind. Zu jeder Messung werden neben der Datensatz-Nummer (Spalte „RecordNo“ in der Datenbank) auch die Kanalbezeichnung sowie das Oszilloskop-Modell genannt:



Übung 3, Messwerte-Visualisierung (c)

Nach der Auswahl einer Oszilloskop-Messung werden die jeweiligen Spannungswerte aus der Datenbank gelesen und in einem Chart auf dem Bildschirm dargestellt:



Quellenverzeichnis

[1] Seite „Relational database“. In: Wikipedia, The Free Encyclopedia. Bearbeitungsstand: 26. März 2023, 22:46 UTC. URL: https://en.wikipedia.org/w/index.php?title=Relational_database&oldid=1146781658. (Abgerufen: 2. Mai 2023)

[2] Seite „SQL“. In: Wikipedia – Die freie Enzyklopädie. Bearbeitungsstand: 24. März 2023, 10:10 UTC. URL: https://de.wikipedia.org/w/index.php?title=SQL&oldid=232133716 (Abgerufen: 2. Mai 2023)

[3] Dave Braunschweig: Datei „Northwind E-R Diagram.png“. In: Wikiversity. Bearbeitungsstand: 20. August 2021, 18:43 UTC. URL: https://en.wikiversity.org/w/index.php?title=File:Northwind_E-R_Diagram.png&oldid=2307920 (Abgerufen: 2. Mai 2023)

[4] Benutzer Cskiran: Datei „Multiple Server .jpg “. In: Wikimedia Commons. Bearbeitungsstand: 2. September 2020, 19:22 UTC. URL: https://commons.wikimedia.org/w/index.php?title=File:Multiple_Server_.jpg&oldid=447144094 (Abgerufen: 2. Mai 2023)

Quellenverzeichnis

[5] Einführung in SQL: „Normalisierung“. Wikibooks, Die freie Bibliothek. Bearbeitungsstand: 17. Januar 2021. URL: https://de.wikibooks.org/w/index.php?title=Einf%C3%BChrung_in_SQL:_Normalisierung&oldid=944511 (Abgerufen: 3. Mai 2023)