



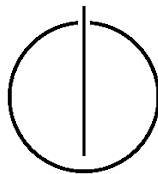
FAKULTÄT FÜR INFORMATIK

DER TECHNISCHEN UNIVERSITÄT MÜNCHEN

Projekt in Datenbanksysteme

Wahlinformationssystem fuer die Bundestagswahl

Korbinian Schmid und Pascal Minnerup



Abstract

Im Rahmen der Vorlesung Datenbanksysteme erstellen die Studenten ein Wahlinformationssystem für die Bundestagswahl. Das Wahlinformationssystem umfasst die Generierung der Ausgangsdaten, das aggregieren und auswerten dieser Daten, sowie die Präsentation auf einer Webseite. Außerdem enthält es ein Benchmarksystem um die Performanz zu testen.

Contents

Abstract	iii
1 Modellierung des Wahlinformationssystems	1
1.1 Informationsstrukturanforderungen	1
1.2 Datenverarbeitungsanforderungen	10
1.3 Vor- und Nachteile Eines DBMS	11
1.4 Integritätsbedingungen	12
1.5 Datenschutzanforderungen	12
2 Vom ER-Modell zum relationalen Schema	13
2.1 Übersetzung der Entities	13
2.1.1 Spezialbehandlung: schwache Entities	13
2.1.2 Spezialbehandlung: Generalisierung	13
2.2 Initial-Entwurf Beziehungen	13
2.3 Verfeinerung des Schemas unter Berücksichtigung der Beziehungen und Kardinalitäten	14
3 SQL-Befehle zur Tabellenerstellung	15
4 SQL-Befehle zur Auswertung der Wahl	19
4.1 Query 1 - Sitzverteilung	19
4.2 Query 2 - Mitglieder des Bundestages	23
4.3 Query 3 - Wahlkreisübersicht	26
4.4 Query 4 - Wahlkreissieger	29
4.5 Query 5 - Überhangmandate	31
4.6 Query 6 - Knappste Sieger	34
4.7 Query 7 - Wahlkreisübersicht (Einzelstimmen)	37
5 Details zur Implementierung der Sitzverteilung	39
6 Benchmark	41
6.1 Queries	41
6.2 Messverfahren und Ziele	41
6.2.1 Temporäre Tabellen	41
6.2.2 Testen der Skalierfähigkeit	42
6.2.3 Technisches	42
6.3 Ergebnisse	43
6.3.1 Q1 - Q6 mit 4 Sekunden Wartezeit	43
6.3.2 Q1 - Q6 mit 2 Sekunden Wartezeit	44

6.3.3	Q1 - Q6 mit 4 Sekunden Wartezeit und WITH-Tables	45
6.3.4	Q1 - Q6 mit 2 Sekunden Wartezeit und WITH-Tables	46
6.3.5	Q7 mit 2 Sekunden Wartezeit	47
6.3.6	Interpretation der Ergebnisse	47
7	Stimmabgabe	49
7.1	Wahlzettel	49
7.2	Auswertung	51
	Bibliography	53

1 Modellierung des Wahlinformationssystems

Das ER Modell 1.1 stellt die verwendeten Klassen der Datenbank für das Wahlinformationssystem dar. Die Klassen enthalten die zunächst zur Verfügung stehenden Daten ohne Vorberechnungen. Aus diesen Daten werden über mehrere Zwischenschritte die Ausgabedaten ermittelt. Die Zwischenschritte werden in eigenen Tabellen abgespeichert. Je nach Konfiguration können diese Tabellen temporäre Tabellen, temporäre Tabellen innerhalb eines SQL Befehls ("with"), materialisierte Anfragen (Materialized Query Tables) oder normale persistente Tabellen sein. Diese Zwischentabellen sind in Abbildung 1.2 dargestellt. Die grünen Boxen stellen die Grundklassen dar, wie sie auch im ER Modell 1.1 enthalten sind. Pfeile stellen Berechnungsbeziehungen da. Der Pfeil von "Erststimmen-NachWahlkreis" nach "Direktmandate" bedeutet zum Beispiel, dass die Direktmandate aus den ErststimmenNachWahlkreis durch aggregation berechnet werden. Die Linien mit Beschriftung "references" stellen eine Fremdschlüsselbeziehung dar. Dementsprechend bedeutet die Verbindung zwischen "Kandidat" und "Direktmandate", dass die Tabelle "Direktmandate" ein Attribut enthält, dass die ID von "Kandidat" referenziert.

1.1 Informationsstrukturanforderungen

Im Folgenden werden die einzelnen Entities anhand ihrer Attribute näher beschrieben. Die Längenangaben sind jeweils in Byte.

Objektbeschreibung: Wahlbezirk

- Anzahl: 25.000
- Attribute:
 - Attributname: ID
 - * Länge: int
 - * Wertebereich: 4
 - * Wiederholungen: 0 ... 25.000
 - * Definiiertheit: 0
 - * Identifizierend: 100%

Objektbeschreibung: Wahlkreis

- Anzahl: 299
- Attribute:

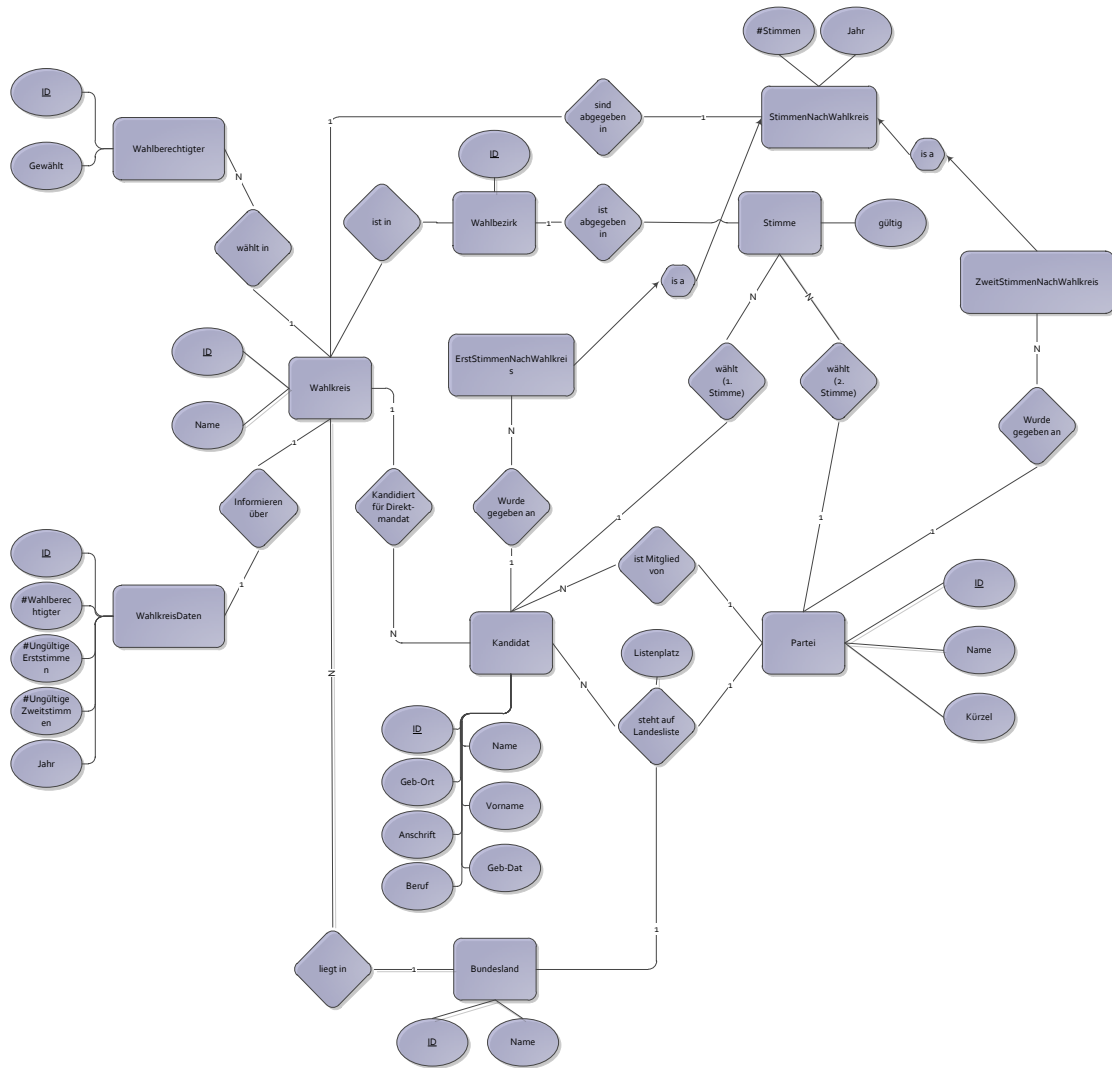


Figure 1.1: ER Modell des Wahlinformationssystems

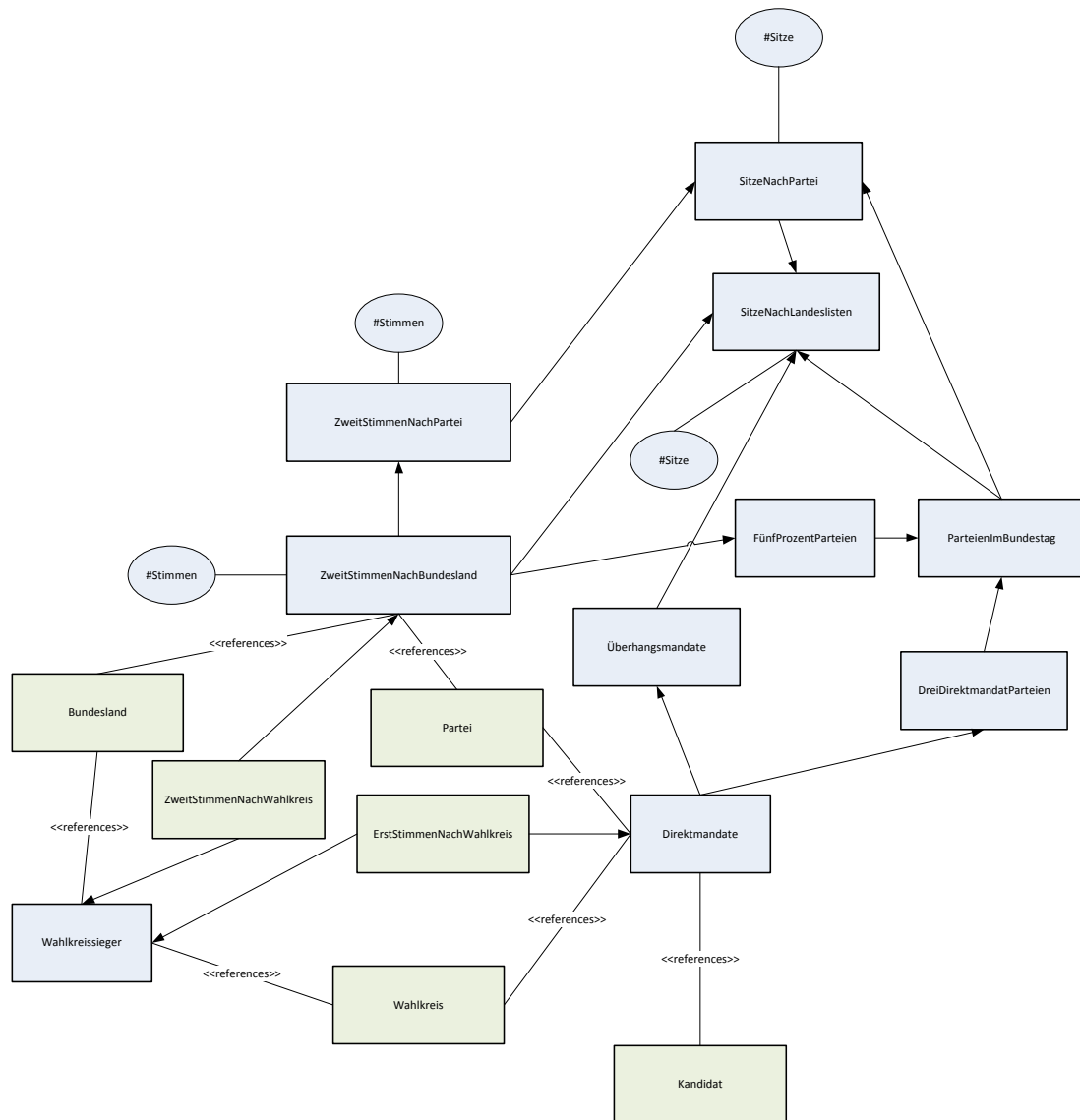


Figure 1.2: Zusammenhang zwischen den Grundtabellen und den temporären Tabellen

- Attributname: ID
 - * Länge: int
 - * Wertebereich: 4
 - * Wiederholungen: 0 ... 299
 - * Definiertheit: 0
 - * Identifizierend: 100%
- Attributname: Wahlkreisname
 - * Länge: char
 - * Wertebereich: 50
 - * Wiederholungen: *
 - * Definiertheit: 0
 - * Identifizierend: 100%

Objektbeschreibung: Bundesland

- Anzahl: 16
- Attribute:
 - Attributname: ID
 - * Länge: int
 - * Wertebereich: 4
 - * Wiederholungen: 0 ... 16
 - * Definiertheit: 0
 - * Identifizierend: 100%
 - Attributname: Name
 - * Länge: char
 - * Wertebereich: 50
 - * Wiederholungen: *
 - * Definiertheit: 0
 - * Identifizierend: 100%

Objektbeschreibung: Kandidat

- Anzahl: 5000
- Attribute:
 - Attributname: ID
 - * Länge: int
 - * Wertebereich: 4

- * Wiederholungen: 0 ... 5000
- * Definiiertheit: 0
- * Identifizierend: 100%
- Attributname: Name
 - * Länge: char
 - * Wertebereich: 50
 - * Wiederholungen: *
 - * Definiiertheit: 0
 - * Identifizierend: 100%
- Attributname: Vorname
 - * Länge: char
 - * Wertebereich: 50
 - * Wiederholungen: *
 - * Definiiertheit: 0
 - * Identifizierend: 100%
- Attributname: Beruf
 - * Länge: char
 - * Wertebereich: 255
 - * Wiederholungen: *
 - * Definiiertheit: 0
 - * Identifizierend: 100%
- Attributname: Anschrift
 - * Länge: char
 - * Wertebereich: 2047
 - * Wiederholungen: *
 - * Definiiertheit: 0
 - * Identifizierend: 100%
- Attributname: Geburtsdatum
 - * Länge: date
 - * Wertebereich: 4
 - * Wiederholungen: *
 - * Definiiertheit: 0
 - * Identifizierend: 100%
- Attributname: Geburtsort
 - * Länge: char

- * Wertebereich: 50
- * Wiederholungen: *
- * Definiertheit: 0
- * Identifizierend: 100%

Objektbeschreibung: Partei

- Anzahl: 50
- Attribute:
 - Attributname: ID
 - * Länge: int
 - * Wertebereich: 4
 - * Wiederholungen: 0 ... 50
 - * Definiertheit: 0
 - * Identifizierend: 100%
 - Attributname: Name
 - * Länge: char
 - * Wertebereich: 50
 - * Wiederholungen: *
 - * Definiertheit: 0
 - * Identifizierend: 100%
 - Attributname: Kürzel
 - * Länge: char
 - * Wertebereich: 50
 - * Wiederholungen: *
 - * Definiertheit: 0
 - * Identifizierend: 100%

Objektbeschreibung: Stimme

- Anzahl: 50.000.000
- Attribute:
 - Attributname: ID
 - * Länge: int
 - * Wertebereich: 4
 - * Wiederholungen: 0 ... 50.000.000
 - * Definiertheit: 0

- * Identifizierend: 100%
- Attributname: Jahr
 - * Länge: int
 - * Wertebereich: 4
 - * Wiederholungen: 1900 ... 2100
 - * Definiiertheit: 0
 - * Identifizierend: 100%

Objektbeschreibung: Wahlberechtigter

- Anzahl: 50.000.000
- Attribute:
 - Attributname: ID
 - * Länge: int
 - * Wertebereich: 4
 - * Wiederholungen: 0 ... 50.000.000
 - * Definiiertheit: 0
 - * Identifizierend: 100%
 - Attributname: Gewählt
 - * Länge: int
 - * Wertebereich: 1
 - * Wiederholungen: 0, 1
 - * Definiiertheit: 0
 - * Identifizierend: 50%

Objektbeschreibung: ErstStimmenNachWahlkreis

- Anzahl: 10.000
- Attribute:
 - Attributname: Anzahl
 - * Länge: int
 - * Wertebereich: 4
 - * Wiederholungen: 0 ... 200.000
 - * Definiiertheit: 0
 - * Identifizierend: 100%
 - Attributname: Jahr
 - * Länge: int

- * Wertebereich: 4
- * Wiederholungen: 1900 ... 2100
- * Definiertheit: 0
- * Identifizierend: 100%

Objektbeschreibung: ZweitStimmenNachWahlkreis

- Anzahl: 10.000
- Attribute:
 - Attributname: Anzahl
 - * Länge: int
 - * Wertebereich: 4
 - * Wiederholungen: 0 ... 200.000
 - * Definiertheit: 0
 - * Identifizierend: 100%
 - Attributname: Jahr
 - * Länge: int
 - * Wertebereich: 4
 - * Wiederholungen: 1900 ... 2100
 - * Definiertheit: 0
 - * Identifizierend: 100%

Objektbeschreibung: WahlkreisDaten

- Anzahl: 10.000
- Attribute:
 - Attributname: ID
 - * Länge: int
 - * Wertebereich: 4
 - * Wiederholungen: 0 ... 10.000
 - * Definiertheit: 0
 - * Identifizierend: 100%
 - Attributname: AnzahlWahlberechtigte
 - * Länge: int
 - * Wertebereich: 4
 - * Wiederholungen: 0 ... 200.000
 - * Definiertheit: 0

- * Identifizierend: 100%
- Attributname: AnzahlUngueltigeErststimmen
 - * Länge: int
 - * Wertebereich: 4
 - * Wiederholungen: 0 ... 200.000
 - * Definiiertheit: 0
 - * Identifizierend: 100%
- Attributname: AnzahlUngueltigeZweitstimmen
 - * Länge: int
 - * Wertebereich: 4
 - * Wiederholungen: 0 ... 200.000
 - * Definiiertheit: 0
 - * Identifizierend: 100%
- Attributname: Jahr
 - * Länge: int
 - * Wertebereich: 4
 - * Wiederholungen: 1900 ... 2100
 - * Definiiertheit: 0
 - * Identifizierend: 100%

Als Beispiel werden hier zwei Beziehungen näher beschrieben:

Beziehungsbeschreibung: wählt (1. Stimme)

- Beteiligte Objekte
 - Stimme als Wähler
 - Kandidat als Empfänger der Stimme
- Anzahl: 45.000.000

Beziehungsbeschreibung: wählt (2. Stimme)

- Beteiligte Objekte
 - Stimme als Wähler
 - Partei als Empfänger der Stimme
- Anzahl: 45.000.000

1.2 Datenverarbeitungsanforderungen

Das Wahlsystem soll unter anderem die folgenden Datenverarbeitungsoperationen unterstützen. Die nachfolgenden Daten sind Schätzungen, die vor der Implementierung erhoben wurden.

Prozessbeschreibung: Berechnung Wahlbeteiligung

- Häufigkeit: jährlich
- benötigte Daten:
 - Stimmen
 - Wahlbezirke
 - Wahlkreise
 - Bundesländer
- Priorität: hoch
- zu verarbeitende Menge
 - 45.000.000 Stimmen
 - 2500 Wahlbezirke
 - 299 Wahlkreise
 - 16 Bundesländer

Prozessbeschreibung: Berechnung Wahlergebnis

- Häufigkeit: jährlich
- benötigte Daten:
 - Stimmen
 - Parteien
 - Kandidaten
 - Wahlbezirke
 - Wahlkreise
 - Landeslisten
- Priorität: hoch
- zu verarbeitende Menge
 - 45.000.000 Stimmen
 - 30 Parteien
 - 2.500 Kandidaten
 - 2.500 Wahlbezirke
 - 299 Wahlkreise

- 500 Landeslisten

Prozessbeschreibung: Berechnung Sitzverteilung

- Häufigkeit: jährlich
- benötigte Daten:
 - Stimmen
 - Parteien
 - Kandidaten
 - Wahlbezirke
 - Wahlkreise
 - Landeslisten
- Priorität: hoch
- zu verarbeitende Menge
 - 50.000.000 Stimmen
 - 30 Parteien
 - 2.500 Kandidaten
 - 2.500 Wahlbezirke
 - 299 Wahlkreise
 - 500 Landeslisten

1.3 Vor- und Nachteile Eines DBMS

Für das Wahlinformationssystem muss entschieden werden, ob ein Datenbank Management System (DBMS) verwendet werden soll. Dafür müssen die Vor- und Nachteile eines solchen abgewägt werden.

Vorteile:

- Verhindern von Redundanz und Inkonsistenz
- Einheitliche Modellierung der Informationen und damit erweiterte Zugriffsmöglichkeiten
- Möglichkeit des Mehrbenutzerbetriebes
- Vorbeugung gegen Datenverlust
- Durchsetzung von Integritätsbedingungen
- Durchsetzung von Sicherheitsmechanismen
- Niedrigere Entwicklungskosten für neue Anwendungsprogramme

Nachteile:

- Kosten für Anschaffung und Betrieb des DBMS
- Gegenüber Papierlösungen: Angreifbarkeit durch Computerviren
- Mögliche Abhängigkeit von einer einzelnen Software

Insgesamt überwiegen die Vorteile eines DBMS.

1.4 Integritätsbedingungen

Die Datenbank sollte folgende Integritätsbedingungen erfüllen:

- Jeder Listenplatz darf maximal einmal vergeben werden.
- Die Funktionalitäten müssen eingehalten werden
- Eine Erststimme kann nur einem Kandidaten gegeben werden, der in dem zugehörigen Wahlkreis für ein Direktmandat kandidiert.
- Eine Zweitstimme kann nur einer Partei gegeben werden, die eine Landesliste in dem zugehörigen Bundesland hat.
- Jede Partei darf in jedem Bundesland maximal eine Landesliste haben.
- Ein Kandidat, der Mitglied einer Partei ist, darf nicht für eine andere Partei kandidieren.

1.5 Datenschutzanforderungen

Um den Datenschutz zu gewährleisten wird als ersten Schritt nach Vorlage des Personalausweises ein Wählpasswort generiert, mit dem der Wahlberechtigte abstimmen kann. Ab dem Zeitpunkt ist seine Stimme entkoppelt von seinem Personalausweis und somit sind Rückschlüsse nicht mehr direkt möglich. Indirekte Rückschlüsse werden dadurch erschwert, dass die Auswertung erst nach Schließung der Wahllokale erfolgt und somit die immer große Stimmzettel-Mengen aggregiert werden.

Zusätzlich zu den Datentechnischen Anforderungen müssen noch mechanische Hindernisse aufgestellt werden. So muss die Stimme im Wahllokal vor Ort abgegeben werden und der Computer darf nicht von außen einsehbar sein. Das Mitnehmen von Fotoapparaten muss unterbunden werden. Die Hardware des Wahlcomputers muss so gestaltet werden, dass Rückschlüsse auf die getroffenen Wahlentscheidungen nicht möglich sind. Zum Beispiel kann der Hauptspeicher klein gewählt werden und nach jeder Stimmabgabe gründlich gelöscht werden.

2 Vom ER-Modell zum relationalen Schema

2.1 Übersetzung der Entities

Wahlberechtigter(ID, Gewählt)

Wahlkreis(ID, Name)

WahlkreisDaten(ID, #Wahlberechtigter, #UngültigeErststimmen, #UngültigeZweitstimmen, Jahr)

Kandidat(ID, Vorname, Nachname, Geburtsdatum, Geburtsort, Anschrift, Beruf)

Partei(ID, Name, Kürzel)

Bundesland(ID, Name)

Stimme(ID, gültig, Jahr)

2.1.1 Spezialbehandlung: schwache Entities

Der *Wahlbezirk* ist eine schwache Entity und hängt von der starken Entity *Wahlkreis* ab:

Wahlbezirk(WahlkreisID, WahlbezirkNr)

2.1.2 Spezialbehandlung: Generalisierung

Die Entity *StimmenNachWahlkreis* ist eine Generalisierung von *ErstStimmenNachWahlkreis* und *ZweitStimmenNachWahlkreis*. Um Joins zu sparen, haben wir daraus zwei anstatt drei Relationen modelliert:

ErstStimmenNachWahlkreis(#Stimmen, Jahr)

ZweitStimmenNachWahlkreis(#Stimmen, Jahr)

2.2 Initial-Entwurf Beziehungen

wählt_in(WahlberechtigterID, WahlkreisID)

informieren_über(WahlkreisDatenID, WahlkreisID)

ist_in(WahlbezirkID, WahlkreisID)

ist_abgegeben_in(StimmeID, WahlbezirkID)

sind_abgegeben_in1(#ErstStimmen, Jahr, WahlkreisID)

sind_abgegeben_in2(#ZweitStimmen, Jahr, WahlkreisID)

liegt_in(WahlkreisID, BundeslandID)

kandidiert_für_Direktmandat(KandidatID, WahlkreisID)
steht_auf_Landesliste(KandidatID, BundeslandID, ParteiID, Listenplatz)
ist_Mitglied_von(KandidatID, ParteiID)
wurde_abgegeben_an1(#ErstStimmen, Jahr, KandidatID)
wurde_abgegeben_an2(#ZweitStimmen, Jahr, ParteiID)
wählt1(StimmeID, KandidatID)
wählt2(StimmeID, ParteiID)

2.3 Verfeinerung des Schemas unter Berücksichtigung der Beziehungen und Kardinalitäten

Wahlberechtigter(ID, Gewählt, WahlkreisID)
Wahlkreis(ID, Name, BundeslandID)
WahlkreisDaten(ID, #Wahlberechtigter, #UngültigeErststimmen, #UngültigeZweitstimmen, Jahr, WahlkreisID)

Das Attribut Listenplatz von *steht_auf_Landesliste* wird zum Attribut von Kandidat:
Kandidat(ID, Vorname, Nachname, Geburtsdatum, Geburtsort, Anschrift, Beruf, ParteiID, BundeslandID, WahlkreisID, ParteiID, Listenplatz)
Partei(ID, Name, Kürzel)
Bundesland(ID, Name)
Wahlbezirk(WahlkreisID, WahlbezirkNr)
Stimme(ID, gültig, Jahr, KandidatID, ParteiID, WahlkreisID, WahlbezirkNr)

Zusammen mit ihrem Fremdschlüssel erhalten die beiden Wahlkreis-aggregierten Stimm-Relationen ihren Schlüssel:

ErstStimmenNachWahlkreis(KandidatID, #Stimmen, Jahr)
ZweitStimmenNachWahlkreis(ParteiID, #Stimmen, Jahr)

3 SQL-Befehle zur Tabellenerstellung

Die SQL-Tabellen können jederzeit neu erzeugt werden durch Verwendung der folgenden Befehle:

```
CREATE TABLE BUNDESLAND (  
    ID BIGINT NOT NULL,  
    NAME VARCHAR (255) NOT NULL,  
    CONSTRAINT CC1288606507352 PRIMARY KEY ( ID) ) ;
```

```
CREATE TABLE WAHLKREIS (  
    ID BIGINT NOT NULL,  
    BUNDESLANDID BIGINT NOT NULL,  
    NAME VARCHAR (255),  
    CONSTRAINT CC1288606603901 PRIMARY KEY ( ID),  
    CONSTRAINT CC1288606617285 FOREIGN KEY (BUNDESLANDID)  
    REFERENCES BUNDESLAND (ID) ON DELETE NO ACTION ON UPDATE NO  
    ACTION ENFORCED ENABLE QUERY OPTIMIZATION )  
    ORGANIZE BY DIMENSIONS ( BUNDESLANDID) ;
```

```
CREATE TABLE WAHLBEZIRK (  
    ID BIGINT NOT NULL,  
    WAHLKREISID BIGINT NOT NULL,  
    CONSTRAINT CC1288606788792 PRIMARY KEY ( ID, WAHLKREISID),  
    CONSTRAINT CC1288606799462 FOREIGN KEY (WAHLKREISID)  
    REFERENCES WAHLKREIS (ID) ON DELETE NO ACTION ON UPDATE NO  
    ACTION ENFORCED ENABLE QUERY OPTIMIZATION )  
    ORGANIZE BY DIMENSIONS ( WAHLKREISID) ;
```

```
CREATE TABLE PARTEI (  
    ID BIGINT NOT NULL,  
    NAME VARCHAR (255),  
    KUERZEL VARCHAR (63) NOT NULL,  
    CONSTRAINT CC1288606983948 PRIMARY KEY ( ID) );
```

```
CREATE TABLE KANDIDAT (  
    ID BIGINT NOT NULL GENERATED ALWAYS AS IDENTITY (START WITH  
    0, INCREMENT BY 1, NO CACHE ),  
    PARTEIID BIGINT,  
    BUNDESLANDID BIGINT,DMWAHLKREISID BIGINT,DMPARTEIID  
    BIGINT ,NACHNAME VARCHAR (255) NOT NULL,  
    VORNAME VARCHAR (255) NOT NULL,
```

```
BERUF VARCHAR (255), GEBURTSDATUM DATE,  
GEBURTSORT VARCHAR (255),ANSCHRIFT VARCHAR (2047),  
LISTENPLATZ INTEGER,CONSTRAINT CC1288607383356 PRIMARY KEY  
( ID),  
CONSTRAINT CC1288607389830 FOREIGN KEY (PARTEIID) REFERENCES  
PARTEI (ID) ON DELETE NO ACTION ON UPDATE NO ACTION ENFORCED  
ENABLE QUERY OPTIMIZATION,  
CONSTRAINT CC1288607385362 FOREIGN KEY (DMWAHLKREISID)  
REFERENCES WAHLKREIS (ID) ON DELETE NO ACTION ON UPDATE NO  
ACTION ENFORCED ENABLE QUERY OPTIMIZATION,  
CONSTRAINT CC1288607388564 FOREIGN KEY (DMPARTEIID)  
REFERENCES PARTEI (ID) ON DELETE NO ACTION ON UPDATE NO  
ACTION ENFORCED ENABLE QUERY OPTIMIZATION,  
CONSTRAINT CC1288607382330 FOREIGN KEY (BUNDESLANDID)  
REFERENCES BUNDESLAND (ID) ON DELETE NO ACTION ON UPDATE NO  
ACTION ENFORCED ENABLE QUERY OPTIMIZATION )  
ORGANIZE BY DIMENSIONS ( PARTEIID, BUNDESLANDID) ;
```

```
CREATE TABLE STIMME (  
ID BIGINT NOT NULL GENERATED ALWAYS AS IDENTITY (START WITH  
0, INCREMENT BY 1, NO CACHE ),  
KandidatID BIGINT,  
ParteiID BIGINT,  
WahlbezirkID BIGINT NOT NULL,  
WahlkreisID BIGINT NOT NULL,  
Jahr INTEGER,  
CONSTRAINT CC1288610679447 PRIMARY KEY ( ID),  
CONSTRAINT CC1288610689165 FOREIGN KEY ( KandidatID)  
REFERENCES KANDIDAT (ID) ON DELETE NO ACTION ON UPDATE NO  
ACTION ENFORCED ENABLE QUERY OPTIMIZATION,  
CONSTRAINT CC1288610702435 FOREIGN KEY (PARTEIID) REFERENCES  
PARTEI (ID) ON DELETE NO ACTION ON UPDATE NO ACTION ENFORCED  
ENABLE QUERY OPTIMIZATION )  
ORGANIZE BY DIMENSIONS ( KANDIDATID, WAHLBEZIRKID,  
WAHLKREISID) ;
```

```
CREATE TABLE ERSTSTIMMENNACHWAHLKREIS (  
KANDIDATID BIGINT NOT NULL,  
WAHLKREISID BIGINT NOT NULL,  
JAHR INTEGER NOT NULL,  
ANZAHL INTEGER NOT NULL,  
CONSTRAINT CC1288610976900 PRIMARY KEY ( KANDIDATID,  
WAHLKREISID, JAHR),  
CONSTRAINT CC1288610994045 FOREIGN KEY (WAHLKREISID)  
REFERENCES WAHLKREIS (ID) ON DELETE NO ACTION ON UPDATE NO  
ACTION ENFORCED ENABLE QUERY OPTIMIZATION )
```

```

    ORGANIZE BY DIMENSIONS ( KANDIDATID, WAHLKREISID) ;
COMMENT ON TABLE ERSTSTIMMENNACHWAHLKREIS IS 'Wahlergebnis_1._Stimme';

CREATE TABLE ZWEITSTIMMENNACHWAHLKREIS (
    PARTEIID BIGINT NOT NULL,
    WAHLKREISID BIGINT NOT NULL,JAHR INTEGER NOT NULL,
    ANZAHL INTEGER NOT NULL,
    CONSTRAINT CC1288610976900 PRIMARY KEY ( PARTEIID,
    WAHLKREISID, JAHR),
    CONSTRAINT CC1288610983160 FOREIGN KEY (PARTEIID) REFERENCES
    PARTEI (ID) ON DELETE NO ACTION ON UPDATE NO ACTION ENFORCED
    ENABLE QUERY OPTIMIZATION,
    CONSTRAINT CC1288610994045 FOREIGN KEY (WAHLKREISID)
    REFERENCES WAHLKREIS (ID) ON DELETE NO ACTION ON UPDATE NO
    ACTION ENFORCED ENABLE QUERY OPTIMIZATION )
    ORGANIZE BY DIMENSIONS ( PARTEIID, WAHLKREISID) ;
COMMENT ON TABLE ZWEITSTIMMENNACHWAHLKREIS IS 'Wahlergebnis_2._Stimme';

CREATE TABLE Wahlberechtigter (
    ID BIGINT PRIMARY KEY GENERATED ALWAYS AS IDENTITY (START
    WITH 0, INCREMENT BY 1, NO CACHE ),
    WahlkreisID BIGINT REFERENCES Wahlkreis ,
    WahlbezirkID BIGINT,
    Gewaehlt INTEGER WITH DEFAULT 0);

CREATE INDEX WAHLKREISIDINDEX ON KORBI.WAHLBERECHTIGTER
(WAHLKREISID ASC) PCTFREE 10 ALLOW REVERSE SCANS PAGE SPLIT
SYMMETRIC COLLECT SAMPLED DETAILED STATISTICS ;

CREATE TABLE WahlkreisDaten (
    WahlkreisID BIGINT NOT NULL PRIMARY KEY REFERENCES
    Wahlkreis ,
    AnzahlWahlberechtigte BIGINT,
    AnzahlUngueltigeErststimmen BIGINT,
    AnzahlUngueltigeZweitstimmen BIGINT,
    Jahr BIGINT);

CREATE TABLE SessionIDs (
    ID VARCHAR (128) PRIMARY KEY NOT NULL,
    WahlkreisID BIGINT NOT NULL REFERENCES Wahlkreis ,
    WahlbezirkID BIGINT NOT NULL);

CREATE TABLE ZUFALLSZAHLENDIREKTMANDATE (
    ZEILE BIGINT NOT NULL GENERATED ALWAYS AS IDENTITY (START
    WITH 0, INCREMENT BY 1, NO CACHE ),
    ZAHL BIGINT NOT NULL);

```

```
CREATE TABLE ZUFALLSZAHLENSitzeNachPartei (  
    ZEILE BIGINT NOT NULL GENERATED ALWAYS AS IDENTITY (START  
    WITH 0, INCREMENT BY 1, NO CACHE ),  
    ZAHL BIGINT NOT NULL);
```

```
CREATE TABLE ZUFALLSZAHLENSitzeNachLandeslisten (  
    ZEILE BIGINT NOT NULL GENERATED ALWAYS AS IDENTITY (START  
    WITH 0, INCREMENT BY 1, NO CACHE ),  
    ZAHL BIGINT NOT NULL);
```


4 SQL-Befehle zur Auswertung der Wahl

In der Implementierung des Wahlinformationssystem sind die Queries modular aufgebaut. Das bedeutet, sie greifen auf dieselben Tabellen zu, die zentral erstellt werden. Um das für dieses Projekt gewünschte Verhalten zu erzielen, werden die zentralen Tabellen für jedes Query neu erzeugt. Im Folgenden sind für jedes Query die SQL statements vermerkt, die zur Berechnung dieses Queries verwendet wurden. Zwischen den Queries gibt es dabei eine Überlappung, also Untertabellen, die in mehreren Queries verwendet werden. Außerdem sind teilweise Zahlen direkt angegeben, die im Programm als Parameter übergeben werden können. Insbesondere gilt dies für 2009 und 2005 für die Wahljahre.

4.1 Query 1 - Sitzverteilung

Stellt die Sitzverteilung im Bundestag als Tortendiagramm und tabellarisch (Partei, Anzahl der Sitze) dar.

WITH

```
ZweitStimmenNachBundesland AS (  
    SELECT w2.ParteiID , wk.BundeslandID , sum(w2.Anzahl) as  
        AnzahlStimmen  
    FROM ZweitStimmenNachWahlkreis w2, Wahlkreis wk  
    WHERE w2.WahlkreisID = wk.ID  
        AND w2.Jahr = 2009  
    GROUP BY wk.BundeslandID , w2.ParteiID ),
```

```
ZweitStimmenNachPartei AS (  
    SELECT ParteiID , SUM(AnzahlStimmen) AS AnzahlStimmen  
    FROM ZweitStimmenNachBundesland  
    GROUP BY ParteiID ),
```

```
MaxErststimmenNachWahlkreis AS (  
    SELECT v.WahlkreisID , MAX(v.Anzahl) AS MaxStimmen  
    FROM ErstStimmenNachWahlkreis v  
    WHERE v.Jahr = 2009  
    GROUP BY v.WahlkreisID ),
```

```
DirektmandateNummer AS (  
    SELECT e.KandidatID , e.WahlkreisID , Row_Number() OVER  
        (PARTITION BY e.WahlkreisID) AS Nummer  
    FROM MaxErststimmenNachWahlkreis m,
```

```
ErstStimmenNachWahlkreis e
WHERE e.WahlkreisID = m.WahlkreisID
      AND m.MaxStimmen = e.Anzahl
      AND e.Jahr = 2009),
```

```
DirektmandateMaxNummer AS (
  SELECT WahlkreisID, MAX(Nummer) AS kMaxNummer
  FROM DirektmandateNummer
  GROUP BY WahlkreisID),
```

```
Direktmandate AS (
  SELECT n.KandidatID, k.ParteiID, k.DMWahlkreisID
  FROM DirektmandateNummer n, DirektmandateMaxNummer mn,
       ZufallsZahlenDirektmandate z, Kandidat k
  WHERE n.WahlkreisID = mn.WahlkreisID
        AND k.ID = n.KandidatID
        AND z.Zeile = mod(n.WahlkreisID, ( SELECT COUNT(*) FROM
       ZufallsZahlenDirektmandate))
        AND n.Nummer = mod(z.Zahl, mn.kMaxNummer) + 1),
```

```
FuenfProzentParteien AS (
  SELECT z.ParteiID
  FROM ZweitStimmenNachPartei z
  GROUP BY z.ParteiID
  HAVING CAST(SUM(z.AnzahlStimmen) AS FLOAT) / ( SELECT
  SUM(AnzahlStimmen) FROM ZweitStimmenNachPartei) >= 0.05),
```

```
DreiDirektMandatParteien AS (
  SELECT dm.ParteiID
  FROM Direktmandate dm
  GROUP BY dm.ParteiID
  HAVING COUNT(*) >= 3),
```

```
ParteienImBundestag AS (
  SELECT *
  FROM FuenfProzentParteien

  UNION

  SELECT *
  FROM DreiDirektMandatParteien),
```

```
Divisoren AS (
  SELECT (ROWNUMBER() OVER (order by w.ID) - 0.5) as Wert
  FROM Wahlkreis w
```

UNION

```
SELECT (ROWNUMBER() OVER (order by w.ID) + (
SELECT COUNT(*)
FROM Wahlkreis) - 0.5) AS Wert
FROM Wahlkreis w),
```

```
ZugriffsreihenfolgeSitzeNachPartei AS (
SELECT p.ParteiID, z.AnzahlStimmen, (z.AnzahlStimmen /
d.wert) as DivWert, ROWNUMBER() OVER (ORDER BY
(z.AnzahlStimmen / d.wert) DESC, rnd.Zahl DESC) as Rang
FROM ParteienImBundestag p, ZweitStimmenNachPartei z,
Divisoren d, ZufallsZahlenSitzeNachPartei rnd
WHERE p.ParteiID = z.ParteiID
AND rnd.Zeile = MOD(p.ParteiID + (d.wert / 1)*( SELECT
COUNT(*) FROM Partei), ( SELECT COUNT(*) FROM ZufallsZa
```

```
SitzeNachPartei AS (
SELECT ParteiID, COUNT(Rang) as AnzahlSitze
FROM ZugriffsreihenfolgeSitzeNachPartei
WHERE Rang <= 2*( SELECT COUNT(*) FROM Wahlkreis)
GROUP BY ParteiID),
```

```
ZugriffsreihenfolgeSitzeNachLandeslisten AS (
SELECT p.ParteiID, z.BundeslandID, z.AnzahlStimmen,
(z.AnzahlStimmen / d.wert) as DivWert, ROWNUMBER() OVER
(PARTITION BY p.ParteiID ORDER BY (z.AnzahlStimmen / d.wert)
DESC, rnd.Zahl DESC) as Rang
FROM ParteienImBundestag p, ZweitStimmenNachBundesland z,
Divisoren d, ZufallsZahlenSitzeNachLandeslisten rnd
WHERE p.ParteiID = z.ParteiID
AND rnd.Zeile = MOD(p.ParteiID + ( SELECT COUNT(*) FROM
Partei)*(z.BundeslandID + ( SELECT COUNT(*) FROM Bunde
```

```
SitzeNachLandeslisten AS (
SELECT z.ParteiID, BundeslandID, COUNT(Rang) as AnzahlSitze
FROM ZugriffsreihenfolgeSitzeNachLandeslisten z,
SitzeNachPartei s
WHERE z.ParteiID = s.ParteiID
AND z.Rang <= s.AnzahlSitze
GROUP BY z.ParteiID, z.BundeslandID, s.ParteiID),
```

```
DirektMandateProParteiUndBundesland AS (
SELECT k.ParteiID, w.BundeslandID, COUNT(*) AS
AnzahlDirektmandate
FROM Direktmandate dm, Kandidat k, Wahlkreis w
```

```
WHERE dm.KandidatID = k.ID
      AND w.ID = k.DMWahlkreisID
GROUP BY k.ParteiID , w.BundeslandID),
```

```
Ueberhangsmandate AS (
  SELECT b.ID AS BundeslandID, b.Name, p.ID AS ParteiID ,
         p.Kuerzel , dmpb.AnzahlDirektmandate - s.AnzahlSitze AS
         AnzahlUeberhangsmandate
  FROM DirektMandateProParteiUndBundesland dmpb,
       SitzeNachLandeslisten s, Partei p, Bundesland b
  WHERE dmpb.BundeslandID = s.BundeslandID
        AND dmpb.ParteiID = s.ParteiID
        AND dmpb.ParteiID = p.ID
        AND dmpb.BundeslandID = b.ID
        AND dmpb.AnzahlDirektmandate - s.AnzahlSitze > 0),
```

```
SumUeberhang AS (
  SELECT ParteiID , SUM(AnzahlUeberhangsmandate) AS
         AnzahlUeberhangsmandate
  FROM Ueberhangsmandate
  GROUP BY ParteiID )
SELECT p.Kuerzel , (s.AnzahlSitze + COALESCE(u.AnzahlUeberhangsmandate , 0)) AS
FROM Partei p, SitzeNachPartei s LEFT OUTER JOIN
      SumUeberhang u ON s.ParteiID = u.ParteiID
WHERE p.ID = s.ParteiID
```

4.2 Query 2 - Mitglieder des Bundestages

Stellt alle Mitglieder des Bundestages als Liste dar.

WITH

```
ZweitStimmenNachBundesland AS (  
    SELECT w2.ParteiID, wk.BundeslandID, sum(w2.Anzahl) as  
        AnzahlStimmen  
    FROM ZweitStimmenNachWahlkreis w2, Wahlkreis wk  
    WHERE w2.WahlkreisID = wk.ID  
        AND w2.Jahr = 2009  
    GROUP BY wk.BundeslandID, w2.ParteiID),
```

```
ZweitStimmenNachPartei AS (  
    SELECT ParteiID, SUM(AnzahlStimmen) AS AnzahlStimmen  
    FROM ZweitStimmenNachBundesland  
    GROUP BY ParteiID),
```

```
MaxErststimmenNachWahlkreis AS (  
    SELECT v.WahlkreisID, MAX(v.Anzahl) AS MaxStimmen  
    FROM ErstStimmenNachWahlkreis v  
    WHERE v.Jahr = 2009  
    GROUP BY v.WahlkreisID),
```

```
DirektmandateNummer AS (  
    SELECT e.KandidatID, e.WahlkreisID, Row_Number() OVER  
        (PARTITION BY e.WahlkreisID) AS Nummer  
    FROM MaxErststimmenNachWahlkreis m,  
        ErstStimmenNachWahlkreis e  
    WHERE e.WahlkreisID = m.WahlkreisID  
        AND m.MaxStimmen = e.Anzahl  
        AND e.Jahr = 2009),
```

```
DirektmandateMaxNummer AS (  
    SELECT WahlkreisID, MAX(Nummer) AS kMaxNummer  
    FROM DirektmandateNummer  
    GROUP BY WahlkreisID),
```

```
Direktmandate AS (  
    SELECT n.KandidatID, k.ParteiID, k.DMWahlkreisID  
    FROM DirektmandateNummer n, DirektmandateMaxNummer mn,  
        ZufallsZahlenDirektmandate z, Kandidat k  
    WHERE n.WahlkreisID = mn.WahlkreisID  
        AND k.ID = n.KandidatID  
        AND z.Zeile = mod(n.WahlkreisID, ( SELECT COUNT(*) FROM
```

```
ZufallsZahlenDirektmandate))  
AND n.Nummer = mod(z.Zahl, mn.kMaxNummer) + 1),
```

```
FuenfProzentParteien AS (  
    SELECT z.ParteiID  
    FROM ZweitStimmenNachPartei z  
    GROUP BY z.ParteiID  
    HAVING CAST(SUM(z.AnzahlStimmen) AS FLOAT) / ( SELECT  
        SUM(AnzahlStimmen) FROM ZweitStimmenNachPartei) >= 0.05),
```

```
DreiDirektMandatParteien AS (  
    SELECT dm.ParteiID  
    FROM Direktmandate dm  
    GROUP BY dm.ParteiID  
    HAVING COUNT(*) >= 3),
```

```
ParteienImBundestag AS (  
    SELECT *  
    FROM FuenfProzentParteien  
  
    UNION  
  
    SELECT *  
    FROM DreiDirektMandatParteien),
```

```
Divisoren AS (  
    SELECT (ROWNUMBER() OVER (order by w.ID) - 0.5) as Wert  
    FROM Wahlkreis w  
  
    UNION  
  
    SELECT (ROWNUMBER() OVER (order by w.ID) + (  
        SELECT COUNT(*)  
        FROM Wahlkreis) - 0.5) AS Wert  
    FROM Wahlkreis w),
```

```
ZugriffsreihenfolgeSitzeNachPartei AS (  
    SELECT p.ParteiID, z.AnzahlStimmen, (z.AnzahlStimmen /  
        d.wert) as DivWert, ROWNUMBER() OVER (ORDER BY  
        (z.AnzahlStimmen / d.wert) DESC, rnd.Zahl DESC) as Rang  
    FROM ParteienImBundestag p, ZweitStimmenNachPartei z,  
        Divisoren d, ZufallsZahlenSitzeNachPartei rnd  
    WHERE p.ParteiID = z.ParteiID  
        AND rnd.Zeile = MOD(p.ParteiID + (d.wert / 1)*( SELECT  
            COUNT(*) FROM Partei), ( SELECT COUNT(*) FROM ZufallsZahlenSitzeNachPartei)
```

```
SitzeNachPartei AS (
    SELECT ParteiID, COUNT(Rang) as AnzahlSitze
    FROM ZugriffsreihenfolgeSitzeNachPartei
    WHERE Rang <= 2*( SELECT COUNT(*) FROM Wahlkreis)
    GROUP BY ParteiID),
```

```
ListenKandidaten AS (
    SELECT ID
    FROM Kandidat
    WHERE BundeslandID IS NOT NULL EXCEPT
    SELECT KandidatID
    FROM Direktmandate ),
```

```
ListenKandidatenMitRang AS (
    SELECT lk.ID, k.ParteiID, b.ID AS BundeslandID, ROWNUMBER()
        OVER (PARTITION BY b.ID, k.ParteiID ORDER BY k.Listenplatz)
        AS Rang
    FROM ListenKandidaten lk, Bundesland b, Kandidat k
    WHERE lk.ID = k.ID
        AND k.BundeslandID = b.ID ),
```

```
Abgeordnete AS (
    SELECT KandidatID
    FROM Direktmandate
```

```
    UNION
```

```
    SELECT lkr.ID
    FROM ListenKandidatenMitRang lkr, SitzeNachLandeslisten s
    WHERE s.ParteiID = lkr.ParteiID
        AND s.BundeslandID = lkr.BundeslandID
        AND lkr.Rang <= s.AnzahlSitze - ( SELECT COUNT(*) FROM
            Direktmandate dm, Wahlkreis w WHERE dm.ParteiID = lkr.ID
        AND dm.DMWahlkreisID = w.ID
        AND w.BundeslandID = lkr.BundeslandID ) )
    SELECT k.Vorname, k.Nachname, p.Kuerzel
    FROM Abgeordnete a, Kandidat k LEFT OUTER JOIN Partei p ON
        k.ParteiID = p.ID
    WHERE a.KandidatID = k.ID
    ORDER BY k.Vorname, k.Nachname
```

4.3 Query 3 - Wahlkreisübersicht

Stellt für einen zufällig ausgewählten Wahlkreis folgende Informationen dar:

1. die Wahlbeteiligung
2. den gewählten Direktkandidaten
3. die prozentuale und absolute Anzahl an Stimmen für jede Partei
4. die Entwicklung der Stimmen im Vergleich zum Vorjahr

```
SELECT Name
FROM Wahlkreis
WHERE ID = 215
```

```
SELECT (1.0 * sum(w2.Anzahl) / max(wd.AnzahlWahlberechtigte)) as Wahlbeteiligung
FROM WahlkreisDaten wd, ZweitStimmenNachWahlkreis w2
WHERE wd.WahlkreisID = w2.WahlkreisID
      AND wd.Jahr = w2.jahr
      AND wd.Jahr = 2009
      AND wd.WahlkreisID = 215
GROUP BY w2.WahlkreisID
```

WITH

```
ErstStimmenEinWahlkreis AS (
    SELECT *
    FROM ErstStimmenNachWahlkreis
    WHERE WahlkreisID=215),
```

```
MaxErststimmenNachWahlkreis AS (
    SELECT v.WahlkreisID, MAX(v.Anzahl) AS MaxStimmen
    FROM ErstStimmenEinWahlkreis v
    WHERE v.Jahr = 2009
    GROUP BY v.WahlkreisID),
```

```
DirektmandateNummer AS (
    SELECT e.KandidatID, e.WahlkreisID, Row_Number() OVER
        (PARTITION BY e.WahlkreisID) AS Nummer
    FROM MaxErststimmenNachWahlkreis m,
        ErstStimmenNachWahlkreis e
    WHERE e.WahlkreisID = m.WahlkreisID
          AND m.MaxStimmen = e.Anzahl
          AND e.Jahr = 2009),
```

```
DirektmandateMaxNummer AS (
```

```

SELECT WahlkreisID , MAX(Nummer) AS kMaxNummer
FROM DirektmandateNummer
GROUP BY WahlkreisID ),

Direktmandate AS (
    SELECT n.KandidatID , k.ParteiID , k.DMWahlkreisID
    FROM DirektmandateNummer n, DirektmandateMaxNummer mn,
         ZufallsZahlenDirektmandate z, Kandidat k
    WHERE n.WahlkreisID = mn.WahlkreisID
          AND k.ID = n.KandidatID
          AND z.Zeile = mod(n.WahlkreisID , ( SELECT COUNT(*) FROM
              ZufallsZahlenDirektmandate ))
          AND n.Nummer = mod(z.Zahl , mn.kMaxNummer) + 1)
SELECT k.Vorname, k.Nachname, p.Kuerzel
FROM Direktmandate dm, Kandidat k, Partei p
WHERE dm.KandidatID = k.ID
      AND dm.ParteiID = p.ID
      AND dm.DMWahlkreisID = 215

WITH ZweitStimmenWahlkreis2009 AS (
    SELECT ParteiID , Anzahl
    FROM ZweitStimmenNachWahlkreis
    WHERE WahlkreisID = 215
          AND Jahr = 2009), ZweitStimmenWahlkreis2005 AS (
    SELECT ParteiID , Anzahl
    FROM ZweitStimmenNachWahlkreis
    WHERE WahlkreisID = 215
          AND Jahr = 2005), SummeZweitStimmenWahlkreis2009 AS (
    SELECT SUM(Anzahl) AS Summe
    FROM ZweitStimmenNachWahlkreis
    WHERE WahlkreisID = 215
          AND Jahr = 2009
    GROUP BY WahlkreisID ), SummeZweitStimmenWahlkreis2005 AS (
    SELECT SUM(Anzahl) AS Summe
    FROM ZweitStimmenNachWahlkreis
    WHERE WahlkreisID = 215
          AND Jahr = 2005
    GROUP BY WahlkreisID )
SELECT p.Kuerzel , COALESCE(w2009.Anzahl, 0) AS Absolut2009 ,
    CAST(COALESCE(w2009.Anzahl, 0) AS FLOAT) / ( SELECT Summe
    FROM SummeZweitStimmenWahlkreis2009) AS Prozentual2009 ,
    COALESCE(w2005.Anzahl, 0) AS Absolut2005 ,
    CAST(COALESCE(w2005.Anzahl, 0) AS FLOAT) / ( SELECT Summe
    FROM SummeZweitStimmenWahlkreis2005) AS Prozentual2005 , (COALESCE(w2009
FROM ZweitStimmenWahlkreis2009 w2009 FULL OUTER JOIN
    ZweitStimmenWahlkreis2005 w2005 ON w2009.ParteiID =

```

```
w2005.ParteiID RIGHT OUTER JOIN Partei p ON p.ID  
= w2009.ParteiID  
ORDER BY Absolut2009 DESC, Absolut2005 DESC
```

4.4 Query 4 - Wahlkreissieger

Stellt die Siegerparteien (Erst-/Zweitstimme) auf einer eingefärbten Deutschlandkartedar. Alternativ dürfen Sie auch eine Tabelle verwenden.

WITH

```
MaxZweitStimmenNachWahlkreis AS (  
    SELECT we.WahlkreisID, MAX(we.Anzahl) AS MaxStimmen  
    FROM ZweitStimmenNachWahlkreis we  
    WHERE we.Jahr = 2009  
    GROUP BY WahlkreisID),
```

```
MaxErststimmenNachWahlkreis AS (  
    SELECT v.WahlkreisID, MAX(v.Anzahl) AS MaxStimmen  
    FROM ErstStimmenNachWahlkreis v  
    WHERE v.Jahr = 2009  
    GROUP BY v.WahlkreisID),
```

```
GewinnerZweitstimmen AS (  
    SELECT we.WahlkreisID, we.ParteiID  
    FROM ZweitStimmenNachWahlkreis we,  
         MaxZweitStimmenNachWahlkreis ms  
    WHERE we.WahlkreisID = ms.WahlkreisID  
         AND we.Anzahl = ms.MaxStimmen  
         AND we.Jahr = 2009),
```

```
GewinnerErststimmen AS (  
    SELECT we.WahlkreisID, we.KandidatID  
    FROM ErstStimmenNachWahlkreis we, MaxErststimmenNachWahlkreis ms  
    WHERE we.WahlkreisID = ms.WahlkreisID  
         AND we.Anzahl = ms.MaxStimmen  
         AND we.Jahr = 2009),
```

```
WahlkreisSieger AS (  
    SELECT g1.WahlkreisID, wk.BundeslandID, p1.Kuerzel AS P1,  
         p2.Kuerzel AS P2  
    FROM GewinnerErststimmen g1, GewinnerZweitstimmen g2,  
         Partei p1, Partei p2, Kandidat k, Wahlkreis wk  
    WHERE g1.WahlkreisID = g2.WahlkreisID  
         AND g1.KandidatID = k.ID  
         AND k.ParteiID = p1.ID  
         AND g2.ParteiID = p2.ID  
         AND wk.ID = g1.WahlkreisID)SELECT *  
FROM WahlkreisSieger
```

```
WITH GewinnerErststimmen(BundeslandID, Partei, GewonneneWahlkreise) AS (  
    SELECT BundeslandID, P1, COUNT(*)  
    FROM WahlkreisSieger  
    GROUP BY BundeslandID, P1 ), GewinnerZweitStimmen(BundeslandID,  
    Partei, GewonneneWahlkreise) AS (  
    SELECT BundeslandID, P2, COUNT(*)  
    FROM WahlkreisSieger  
    GROUP BY BundeslandID, P2 ), GewinnerGesamt(BundeslandID,  
    Partei, GewonneneWahlkreise) AS (  
    SELECT g1.BundeslandID, g1.Partei, g1.GewonneneWahlkreise +  
        g2.GewonneneWahlkreise  
    FROM GewinnerErststimmen g1, GewinnerZweitStimmen g2 )  
SELECT g.BundeslandID, g.Partei  
FROM GewinnerGesamt g  
WHERE NOT EXISTS ( SELECT * FROM GewinnerGesamt g0 WHERE  
    g0.BundeslandID = g.BundeslandID  
    AND g0.GewonneneWahlkreise > g.GewonneneWahlkreise )
```

4.5 Query 5 - Überhangmandate

Stellt die Überhangmandate pro Partei und Bundesland dar.

WITH

```
ZweitStimmenNachBundesland AS (  
    SELECT w2.ParteiID, wk.BundeslandID, sum(w2.Anzahl) as  
        AnzahlStimmen  
    FROM ZweitStimmenNachWahlkreis w2, Wahlkreis wk  
    WHERE w2.WahlkreisID = wk.ID  
        AND w2.Jahr = 2009  
    GROUP BY wk.BundeslandID, w2.ParteiID),
```

```
ZweitStimmenNachPartei AS (  
    SELECT ParteiID, SUM(AnzahlStimmen) AS AnzahlStimmen  
    FROM ZweitStimmenNachBundesland  
    GROUP BY ParteiID),
```

```
MaxErststimmenNachWahlkreis AS (  
    SELECT v.WahlkreisID, MAX(v.Anzahl) AS MaxStimmen  
    FROM ErstStimmenNachWahlkreis v  
    WHERE v.Jahr = 2009  
    GROUP BY v.WahlkreisID),
```

```
DirektmandateNummer AS (  
    SELECT e.KandidatID, e.WahlkreisID, Row_Number() OVER  
        (PARTITION BY e.WahlkreisID) AS Nummer  
    FROM MaxErststimmenNachWahlkreis m,  
        ErstStimmenNachWahlkreis e  
    WHERE e.WahlkreisID = m.WahlkreisID  
        AND m.MaxStimmen = e.Anzahl  
        AND e.Jahr = 2009),
```

```
DirektmandateMaxNummer AS (  
    SELECT WahlkreisID, MAX(Nummer) AS kMaxNummer  
    FROM DirektmandateNummer  
    GROUP BY WahlkreisID),
```

```
Direktmandate AS (  
    SELECT n.KandidatID, k.ParteiID, k.DMWahlkreisID  
    FROM DirektmandateNummer n, DirektmandateMaxNummer mn,  
        ZufallsZahlenDirektmandate z, Kandidat k  
    WHERE n.WahlkreisID = mn.WahlkreisID  
        AND k.ID = n.KandidatID  
        AND z.Zeile = mod(n.WahlkreisID, ( SELECT COUNT(*) FROM
```

```
ZufallsZahlenDirektmandate))  
AND n.Nummer = mod(z.Zahl, mn.kMaxNummer) + 1),
```

```
FuenfProzentParteien AS (  
    SELECT z.ParteiID  
    FROM ZweitStimmenNachPartei z  
    GROUP BY z.ParteiID  
    HAVING CAST(SUM(z.AnzahlStimmen) AS FLOAT) / ( SELECT  
        SUM(AnzahlStimmen) FROM ZweitStimmenNachPartei) >= 0.05),
```

```
DreiDirektMandatParteien AS (  
    SELECT dm.ParteiID  
    FROM Direktmandate dm  
    GROUP BY dm.ParteiID  
    HAVING COUNT(*) >= 3),
```

```
ParteienImBundestag AS (  
    SELECT *  
    FROM FuenfProzentParteien  
  
    UNION  
  
    SELECT *  
    FROM DreiDirektMandatParteien),
```

```
Divisoren AS (  
    SELECT (ROWNUMBER() OVER (order by w.ID) - 0.5) as Wert  
    FROM Wahlkreis w  
  
    UNION  
  
    SELECT (ROWNUMBER() OVER (order by w.ID) + (  
        SELECT COUNT(*)  
        FROM Wahlkreis) - 0.5) AS Wert  
    FROM Wahlkreis w),
```

```
ZugriffsreihenfolgeSitzeNachPartei AS (  
    SELECT p.ParteiID, z.AnzahlStimmen, (z.AnzahlStimmen /  
        d.wert) as DivWert, ROWNUMBER() OVER (ORDER BY  
        (z.AnzahlStimmen / d.wert) DESC, rnd.Zahl DESC) as Rang  
    FROM ParteienImBundestag p, ZweitStimmenNachPartei z,  
        Divisoren d, ZufallsZahlenSitzeNachPartei rnd  
    WHERE p.ParteiID = z.ParteiID  
        AND rnd.Zeile = MOD(p.ParteiID + (d.wert / 1)*( SELECT  
            COUNT(*) FROM Partei), ( SELECT COUNT(*) FROM ZufallsZahlenSitzeNachPartei))
```

```
SitzeNachPartei AS (  
    SELECT ParteiID, COUNT(Rang) as AnzahlSitze  
    FROM ZugriffsreihenfolgeSitzeNachPartei  
    WHERE Rang <= 2*( SELECT COUNT(*) FROM Wahlkreis)  
    GROUP BY ParteiID),
```

```
ZugriffsreihenfolgeSitzeNachLandeslisten AS (  
    SELECT p.ParteiID, z.BundeslandID, z.AnzahlStimmen,  
           (z.AnzahlStimmen / d.wert) as DivWert, ROWNUMBER() OVER  
           (PARTITION BY p.ParteiID ORDER BY (z.AnzahlStimmen / d.wert)  
           DESC, rnd.Zahl DESC) as Rang  
    FROM ParteienImBundestag p, ZweitStimmenNachBundesland z,  
         Divisoren d, ZufallsZahlenSitzeNachLandeslisten rnd  
    WHERE p.ParteiID = z.ParteiID  
           AND rnd.Zeile = MOD(p.ParteiID + ( SELECT COUNT(*) FROM  
           Partei)*(z.BundeslandID + ( SELECT COUNT(*) FROM Bunde
```

```
SitzeNachLandeslisten AS (  
    SELECT z.ParteiID, BundeslandID, COUNT(Rang) as AnzahlSitze  
    FROM ZugriffsreihenfolgeSitzeNachLandeslisten z,  
         SitzeNachPartei s  
    WHERE z.ParteiID = s.ParteiID  
           AND z.Rang <= s.AnzahlSitze  
    GROUP BY z.ParteiID, z.BundeslandID, s.ParteiID),
```

```
DirektMandateProParteiUndBundesland AS (  
    SELECT k.ParteiID, w.BundeslandID, COUNT(*) AS  
           AnzahlDirektmandate  
    FROM Direktmandate dm, Kandidat k, Wahlkreis w  
    WHERE dm.KandidatID = k.ID  
           AND w.ID = k.DMWahlkreisID  
    GROUP BY k.ParteiID, w.BundeslandID),
```

```
Ueberhangsmandate AS (  
    SELECT b.ID AS BundeslandID, b.Name, p.ID AS ParteiID,  
           p.Kuerzel, dmpb.AnzahlDirektmandate - s.AnzahlSitze AS  
           AnzahlUeberhangsmandate  
    FROM DirektMandateProParteiUndBundesland dmpb,  
         SitzeNachLandeslisten s, Partei p, Bundesland b  
    WHERE dmpb.BundeslandID = s.BundeslandID  
           AND dmpb.ParteiID = s.ParteiID  
           AND dmpb.ParteiID = p.ID  
           AND dmpb.BundeslandID = b.ID  
           AND dmpb.AnzahlDirektmandate - s.AnzahlSitze > 0)  
  
SELECT *  
FROM Ueberhangsmandate
```

4.6 Query 6 - Knappste Sieger

Stellt die Top 10 der knappsten Sieger für alle Parteien dar. Die knappsten Sieger sind die gewählten Erstkandidaten, welche mit dem geringsten Vorsprung gegenüber ihren Konkurrenten gewonnen haben. Sollte eine Partei keinen Wahlkreis gewonnen haben, sollen stattdessen die Wahlkreise ausgegeben werden, in denen sie am knappsten verloren hat.

WITH

```
MaxStimmen(WahlkreisID , Anzahl) AS (  
    SELECT we.WahlkreisID , MAX(we.Anzahl)  
    FROM ErstStimmenNachWahlkreis we  
    WHERE we.Jahr = 2009  
    GROUP BY we.WahlkreisID ),
```

```
Erster(WahlkreisID , KandidatID , ParteiID , Anzahl) AS (  
    SELECT we.WahlkreisID , we.KandidatID , k.ParteiID , we.Anzahl  
    FROM ErstStimmenNachWahlkreis we, MaxStimmen ms, Kandidat k  
    WHERE we.WahlkreisID = ms.WahlkreisID  
        AND we.Jahr = 2009  
        AND we.KandidatID = k.ID  
        AND we.Anzahl = ms.Anzahl ),
```

```
RestKandidaten(KandidatID) AS (  
    SELECT KandidatID  
    FROM ErstStimmenNachWahlkreis we  
    WHERE we.Jahr = 2009 EXCEPT  
    SELECT KandidatID  
    FROM Erster ),
```

```
MaxStimmenRest(WahlkreisID , Anzahl) AS (  
    SELECT we.WahlkreisID , MAX(we.Anzahl)  
    FROM ErstStimmenNachWahlkreis we, RestKandidaten r  
    WHERE we.KandidatID = r.KandidatID  
        AND we.Jahr = 2009  
    GROUP BY we.WahlkreisID ),
```

```
Zweiter(WahlkreisID , KandidatID , ParteiID , Anzahl) AS (  
    SELECT k.DMWahlkreisID , k.ID , k.ParteiID , we.Anzahl  
    FROM Kandidat k, ErstStimmenNachWahlkreis we,  
        MaxStimmenRest ms  
    WHERE we.Anzahl = ms.Anzahl  
        AND we.WahlkreisID = ms.WahlkreisID  
        AND we.KandidatID = k.ID  
        AND we.Jahr = 2009 ),
```



```
KnappsteSieger(GewinnerID, Differenz, VerliererID,
WahlkreisID) AS (
    SELECT e.ParteiID, e.Anzahl - z.Anzahl AS Differenz,
           z.ParteiID, e.WahlkreisID
    FROM Erster e, Zweiter z
    WHERE e.WahlkreisID = z.WahlkreisID
    ORDER BY e.ParteiID, Differenz ),
```

```
KnappsteSiegerRang(Rang, GewinnerID, Differenz, VerliererID,
WahlkreisID) AS (
    SELECT ROWNUMBER() OVER (PARTITION
    BY kn.GewinnerID ORDER BY
           kn.Differenz), kn.GewinnerID, kn.Differenz, kn.VerliererID,
           kn.WahlkreisID
    FROM KnappsteSieger kn ),
```

```
ParteienOhneSieg(ParteiID) AS (
    SELECT ID
    FROM Partei EXCEPT
    SELECT GewinnerID
    FROM KnappsteSiegerRang ),
```

```
KnappsteSiegerOutput(Rang, Vorname, Nachname, Partei,
Wahlkreis, Differenz, Typ) AS (
    SELECT knr.Rang, k.Vorname, k.Nachname, p.Kuerzel, wk.Name,
           knr.Differenz, 'Gewinner'
    FROM KnappsteSiegerRang knr, Partei p, Kandidat k,
           Wahlkreis wk
    WHERE Rang <= 10
           AND knr.GewinnerID = p.ID
           AND knr.WahlkreisID = k.DMWahlkreisID
           AND k.ParteiID = p.ID
           AND wk.ID = knr.WahlkreisID ),
```

```
KnappsteVerlierer(ParteiID, KandidatID, AbstandZumErsten,
WahlkreisID) AS (
    SELECT pos.ParteiID, k.ID, e.Anzahl - we.Anzahl, w.ID
    FROM ParteienOhneSieg pos, Erster e, Wahlkreis w,
           ErstStimmenNachWahlkreis we, Kandidat k
    WHERE we.WahlkreisID = w.ID
           AND we.Jahr = 2009
           AND e.WahlkreisID = w.ID
           AND we.KandidatID = k.ID
           AND k.DMWahlkreisID = w.ID
           AND k.ParteiID = pos.ParteiID ),
```

```
KnappsteVerliererRang(Rang, ParteiID, KandidatID,
AbstandZumErsten, WahlkreisID) AS (
    SELECT ROWNUMBER() OVER (PARTITION BY kv.ParteiID ORDER BY
        kv.AbstandZumErsten ASC), kv.ParteiID, kv.KandidatID,
        kv.AbstandZumErsten, kv.WahlkreisID
    FROM KnappsteVerlierer kv ),
```

```
KnappsteVerliererOutput(Rang, Vorname, Nachname, Partei,
Wahlkreis, Differenz, Typ) AS (
    SELECT kvr.Rang, k.Vorname, k.Nachname, p.Kuerzel, wk.Name,
        kvr.AbstandZumErsten, 'Verlierer'
    FROM KnappsteVerliererRang kvr, Partei p, Kandidat k,
        Wahlkreis wk
    WHERE Rang <= 10
        AND kvr.ParteiID = p.ID
        AND kvr.WahlkreisID = k.DMWahlkreisID
        AND k.ParteiID = p.ID
        AND wk.ID = kvr.WahlkreisID ),
```

```
GewinnerUndVerliererOutput AS (
    SELECT *
    FROM KnappsteSiegerOutput

    UNION ALL

    SELECT *
    FROM KnappsteVerliererOutput )
SELECT *
FROM GewinnerUndVerliererOutput
ORDER BY Typ
```

4.7 Query 7 - Wahlkreisübersicht (Einzelstimmen)

Diese Anfrage soll die gleichen Informationen wie Q3 darstellen, aber auf Einzelstimmen durchgeführt werden. Die Zufallsauswahl können Sie auf die ersten 5 bayrischen Wahlkreise (213, ..., 217) beschränken.

WITH

```
ErstStimmenEinWahlkreis AS (  
    SELECT *  
    FROM ErstStimmenNachWahlkreis  
    WHERE WahlkreisID=213),
```

```
MaxErststimmenNachWahlkreis AS (  
    SELECT v.WahlkreisID, MAX(v.Anzahl) AS MaxStimmen  
    FROM ErstStimmenEinWahlkreis v  
    WHERE v.Jahr = 2009  
    GROUP BY v.WahlkreisID),
```

```
DirektmandateNummer AS (  
    SELECT e.KandidatID, e.WahlkreisID, Row_Number() OVER  
        (PARTITION BY e.WahlkreisID) AS Nummer  
    FROM MaxErststimmenNachWahlkreis m,  
        ErstStimmenNachWahlkreis e  
    WHERE e.WahlkreisID = m.WahlkreisID  
        AND m.MaxStimmen = e.Anzahl  
        AND e.Jahr = 2009),
```

```
DirektmandateMaxNummer AS (  
    SELECT WahlkreisID, MAX(Nummer) AS kMaxNummer  
    FROM DirektmandateNummer  
    GROUP BY WahlkreisID),
```

```
Direktmandate AS (  
    SELECT n.KandidatID, k.ParteiID, k.DMWahlkreisID  
    FROM DirektmandateNummer n, DirektmandateMaxNummer mn,  
        ZufallsZahlenDirektmandate z, Kandidat k  
    WHERE n.WahlkreisID = mn.WahlkreisID  
        AND k.ID = n.KandidatID  
        AND z.Zeile = mod(n.WahlkreisID, ( SELECT COUNT(*) FROM  
            ZufallsZahlenDirektmandate ))  
        AND n.Nummer = mod(z.Zahl, mn.kMaxNummer) + 1)  
SELECT k.Vorname, k.Nachname, p.Kuerzel  
FROM Direktmandate dm, Kandidat k, Partei p  
WHERE dm.KandidatID = k.ID  
    AND dm.ParteiID = p.ID
```

```
AND dm.DMWahlkreisID = 213

WITH ZweitStimmenWahlkreis2009 AS (
    SELECT ParteiID, Anzahl
    FROM ZweitStimmenNachWahlkreis
    WHERE WahlkreisID = 213
        AND Jahr = 2009),

ZweitStimmenWahlkreis2005 AS (
    SELECT ParteiID, Anzahl
    FROM ZweitStimmenNachWahlkreis
    WHERE WahlkreisID = 213
        AND Jahr = 2005),

SummeZweitStimmenWahlkreis2009 AS (
    SELECT SUM(Anzahl) AS Summe
    FROM ZweitStimmenNachWahlkreis
    WHERE WahlkreisID = 213
        AND Jahr = 2009
    GROUP BY WahlkreisID ),

SummeZweitStimmenWahlkreis2005 AS (
    SELECT SUM(Anzahl) AS Summe
    FROM ZweitStimmenNachWahlkreis
    WHERE WahlkreisID = 213
        AND Jahr = 2005
    GROUP BY WahlkreisID )

SELECT p.Kuerzel, COALESCE(w2009.Anzahl, 0) AS Absolut2009,
    CAST(COALESCE(w2009.Anzahl, 0) AS FLOAT) /
        ( SELECT Summe FROM SummeZweitStimmenWahlkreis2009)
    AS Prozentual2009,
    COALESCE(w2005.Anzahl, 0) AS Absolut2005,
    CAST(COALESCE(w2005.Anzahl, 0) AS FLOAT) /
        (SELECT Summe FROM SummeZweitStimmenWahlkreis2005)
    AS Prozentual2005,
    (COALESCE(w2009.Anzahl, 0) – COALESCE(w2005.Anzahl, 0)) as Aenderung
FROM ZweitStimmenWahlkreis2009 w2009 FULL OUTER JOIN
    ZweitStimmenWahlkreis2005 w2005
    ON w2009.ParteiID = w2005.ParteiID
    RIGHT OUTER JOIN Partei p ON p.ID = w2009.ParteiID
ORDER BY Absolut2009 DESC, Absolut2005 DESC
```

5 Details zur Implementierung der Sitzverteilung

2009 wurde die Sitzverteilung nach dem Verfahren von Sainte-Laguë/Schepers ermittelt. Um dieses Verfahren umzusetzen werden drei verschiedene Algorithmen vorgeschlagen, die rechtlich gleichwertig sind. Für die Implementierung in SQL bietet sich das Höchstzahlverfahren an. In Kapitel 4 ist das Verfahren in folgenden Teiltabellen umgesetzt:

```
SitzeNachPartei AS (  
    SELECT ParteiID, COUNT(Rang) as AnzahlSitze  
    FROM ZugriffsreihenfolgeSitzeNachPartei  
    WHERE Rang <= 2*( SELECT COUNT(*) FROM Wahlkreis)  
    GROUP BY ParteiID), ZugriffsreihenfolgeSitzeNachLandeslisten AS (  
    SELECT p.ParteiID, z.BundeslandID, z.AnzahlStimmen,  
           (z.AnzahlStimmen / d.wert) as DivWert, ROWNUMBER() OVER  
           (PARTITION BY p.ParteiID ORDER BY (z.AnzahlStimmen / d.wert)  
           DESC, rnd.Zahl DESC) as Rang  
    FROM ParteienImBundestag p, ZweitStimmenNachBundesland z,  
         Divisoren d, ZufallsZahlenSitzeNachLandeslisten rnd  
    WHERE p.ParteiID = z.ParteiID  
           AND rnd.Zeile = MOD(p.ParteiID + (SELECT COUNT(*) FROM Partei)  
                               *(z.BundeslandID + ( SELECT COUNT(*) FROM Bundesland)*(  
                               ( SELECT COUNT(*) FROM ZufallsZahlenSitzeNachLandeslist
```

```
SitzeNachPartei AS (  
    SELECT ParteiID, COUNT(Rang) as AnzahlSitze  
    FROM ZugriffsreihenfolgeSitzeNachPartei  
    WHERE Rang <= 2*( SELECT COUNT(*) FROM Wahlkreis)  
    GROUP BY ParteiID), ZugriffsreihenfolgeSitzeNachLandeslisten AS (  
    SELECT p.ParteiID, z.BundeslandID, z.AnzahlStimmen,  
           (z.AnzahlStimmen / d.wert) as DivWert, ROWNUMBER() OVER  
           (PARTITION BY p.ParteiID ORDER BY (z.AnzahlStimmen / d.wert)  
           DESC, rnd.Zahl DESC) as Rang  
    FROM ParteienImBundestag p, ZweitStimmenNachBundesland z,  
         Divisoren d, ZufallsZahlenSitzeNachLandeslisten rnd  
    WHERE p.ParteiID = z.ParteiID  
           AND rnd.Zeile = MOD(p.ParteiID + (SELECT COUNT(*) FROM Partei)  
                               *(z.BundeslandID + ( SELECT COUNT(*) FROM Bundesland)*(  
                               ( SELECT COUNT(*) FROM ZufallsZahlenSitzeNachLandeslist
```

Das im Höchstzahlverfahren vorgesehene Schrittweise erhöhen des Divisors für jede einzelne Partei wird durch die Divisoren Tabelle verwirklicht. Für jede Partei wird für jeden Divisor eine Punktzahl ermittelt. Anschließend werden für die höchsten Punktzahlen Plätze im Bundestag vergeben.

Eine der Schwierigkeiten bei der Umsetzung des Höchstzahlverfahren in SQL ist die Behandlung von Stimmgleichständen. Wenn zwei oder mehr Parteien exakt gleich viele Stimmen haben, muss der Wahlleiter zufällig den Gewinner bestimmen. Die SQL Abfrage hat aber nicht die Möglichkeit während der Berechnung den Wahlleiter zu kontaktieren. Stattdessen muss der Wahlleiter Zufallsentscheidungen für die Maximal mögliche Anzahl von Entscheidungsfällen bereits vor der Auswertung bereitstellen. Diese Zufallsentscheidungen sind in der Implementierung zu diesem Dokument in der Tabelle Zufallszahlen abgelegt. Die Tabelle hat die Form (Zeile, Zahl). Die Zeile entspricht der Instanz der Zufallsentscheidung. Für die 7. mögliche Zufallsentscheidung wird also die Zeile 7 der Zufallstabelle verwendet. Durch automatische Generierung der Spalte Zeile wird sichergestellt, dass jede Zeilenzahl zwischen 0 und $(\#Zeilen - 1)$ genau einmal vorkommt. Die Zahl wird dann als Entscheidungsfinder verwendet: Wenn es n mögliche Alternativen gibt und die Zufallszahl z die Entscheidung liefern soll, so wird die Alternative $z \bmod n$ gewählt.

Dieses Zufallsverfahren kommt für die Ermittlung der Direktmandate, für das Höchstzahlverfahren zur Ermittlung der Sitze pro Partei, sowie für das Höchstzahlverfahren zur Ermittlung der Sitze pro Landesliste zum Einsatz. Um Mehrfachverwendung derselben Zufallszahlen zu vermeiden, werden drei getrennte Zufallszahlentabellen verwendet.

6 Benchmark

6.1 Queries

Zur Durchführung des Benchmarks wurde ein Transaktionsmix von sechs Queries und zugehöriger Aufrufhäufigkeit betrachtet:

- Q1 - Sitzplatzverteilung Bundestag mit Kuchendiagramm (Häufigkeit: 25%)
- Q2 - Liste der Abgeordneten (Häufigkeit: 10%)
- Q3 - Detailergebnisse eines beliebigen Wahlkreises (Häufigkeit: 25%)
- Q4 - Wahlkreissieger mit eingefärbter Deutschlandkarte (Häufigkeit: 10%)
- Q5 - Überhangsmandate (Häufigkeit: 10%)
- Q6 - Knappste Sieger und knappste Verlierer (Häufigkeit: 20%)

Q3 arbeitet auf Wahlkreis-aggregierten Stimmen. Das heißt für jeden Wahlkreis befindet sich für alle darin angetretenen Kandidaten und Parteien bereits die Summe der abgegebenen Erst- bzw. Zweitstimmen als Eintrag in einer Datenbanktabelle.

Um dennoch die Performance des Systems auf Einzelstimmen zu testen, betrachten wir gesondert ein Query Q7, das das gleiche berechnet wie Q3, dem allerdings Einzelstimmen für die Berechnung zugrunde liegen:

- Q7 - Detailergebnisse eines beliebigen Wahlkreises (Einzelstimmen)

Da Q7 über 62 Millionen Tupel aggregieren muss, wurden für den Benchmark aus Laufzeitgründen nur die Stimmen der Wahlkreise 213 - 217 in die Datenbank geladen.

6.2 Messverfahren und Ziele

Neben der Laufzeit der Queries untereinander wurde insbesondere ihre Skalierfähigkeit getestet. Außerdem gab der Benchmark Aufschluss darüber welche temporäre Tabellen-Strategie mehr Performanz aufweist.

6.2.1 Temporäre Tabellen

Alle Queries benutzen für die Berechnung von Zwischenergebnissen temporäre Tabellen. In DB2 gibt es dafür im Wesentlichen zwei Ansätze: Das explizite Erstellen temporärer Tabellen mittels `CREATE GLOBAL TEMPORARY TABLE` oder die Verwendung von Tables, die mittels dem SQL-Befehl `WITH` erstellt wurden. Die wesentlichen Unterschiede der beiden Methoden liegen in ihrer Lebensdauer¹. Um die Performanz der beiden Metho-

¹DB2 Temporary Tables: http://www.cs.newpaltz.edu/~pletcha/DB/db2_TempTables.html

den zu testen wurde der Benchmark für Q1-Q5 einmal unter Verwendung von GLOBAL TEMPORARY TABLES und einmal mittels WITH durchgeführt.

6.2.2 Testen der Skalierfähigkeit

Die Messung wurde mit 1, 2, 5, 7 und 10 parallel aktiven simulierten Browsern durchgeführt. Für die Queries Q1 - Q6 wurde der Benchmark einmal mit zwei Sekunden und einmal mit vier Sekunden Wartezeit zwischen den Queries durchgeführt.

6.2.3 Technisches

Alle simulierten Browser sind in Java geschrieben und stellten alle Anfragen nach ihrer Häufigkeit in zufälliger Reihenfolge an einen lokalen Tomcat-Server. Auf dem Server lief ebenfalls Java-Code der die entsprechenden SQL-Statements generierte und an einen lokalen DB2-Server weiterleitete. Aus der DB2-Antwort wurde dann mittels Servlets eine HTML-Seite generiert und an den simulierten Browser gesendet, der die Zeit zwischen Anfrage und Antwort stoppte. Der Benchmark wurde auf einem Intel Core 2 Duo mit 1.50 GHz und 2.00 GB RAM durchgeführt.

6.3 Ergebnisse

Folgende Diagramme zeigen die Performance der einzelnen Queries unter steigender Anzahl von parallelen Zugriffen. Auf der x-Achse befindet sich die Anzahl der parallel aktiven Browser und auf der y-Achse die dazugehörige durchschnittliche Server-Responsezeit in Millisekunden.

6.3.1 Q1 - Q6 mit 4 Sekunden Wartezeit

Das Benchmark-Ergebnis mit einer Wartezeit von jeweils *vier* Sekunden pro Browser zwischen den Queries:

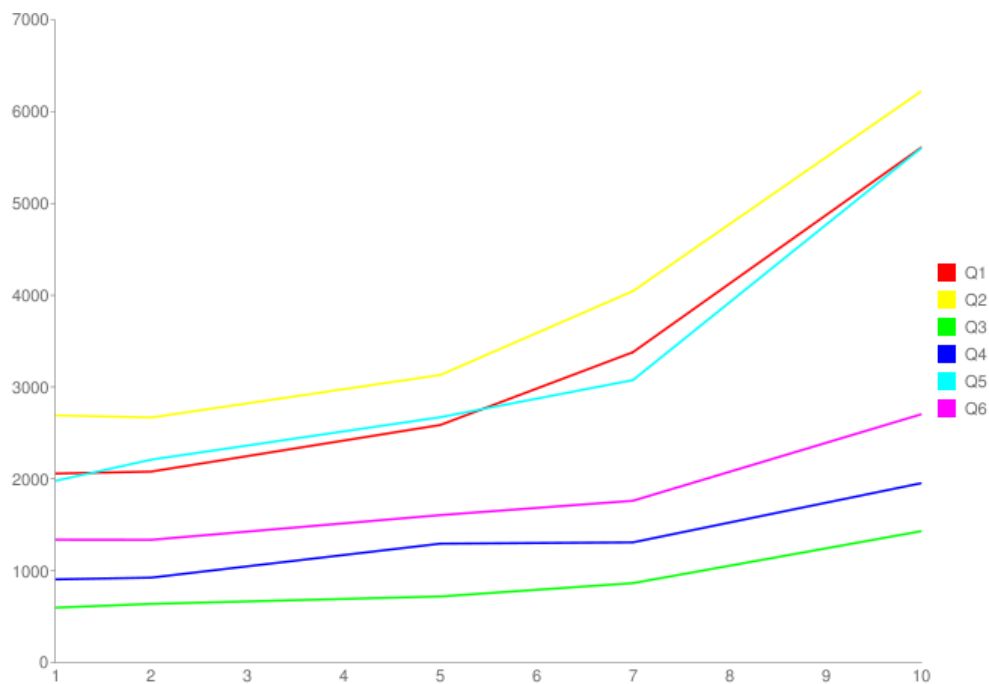


Figure 6.1: Durchschnittliche Response-Time bei einer Wartezeit von 4 Sekunden

Man sieht dass die Berechnung von Q3 (Detailergebnisse Wahlkreis) am schnellsten ist, während die Berechnung von Q2 (Abgeordneten) am längsten dauert.

6.3.2 Q1 - Q6 mit 2 Sekunden Wartezeit

Das Benchmark-Ergebnis mit einer reduzierten Wartezeit von jeweils *zwei* Sekunden pro Browser zwischen den Queries:

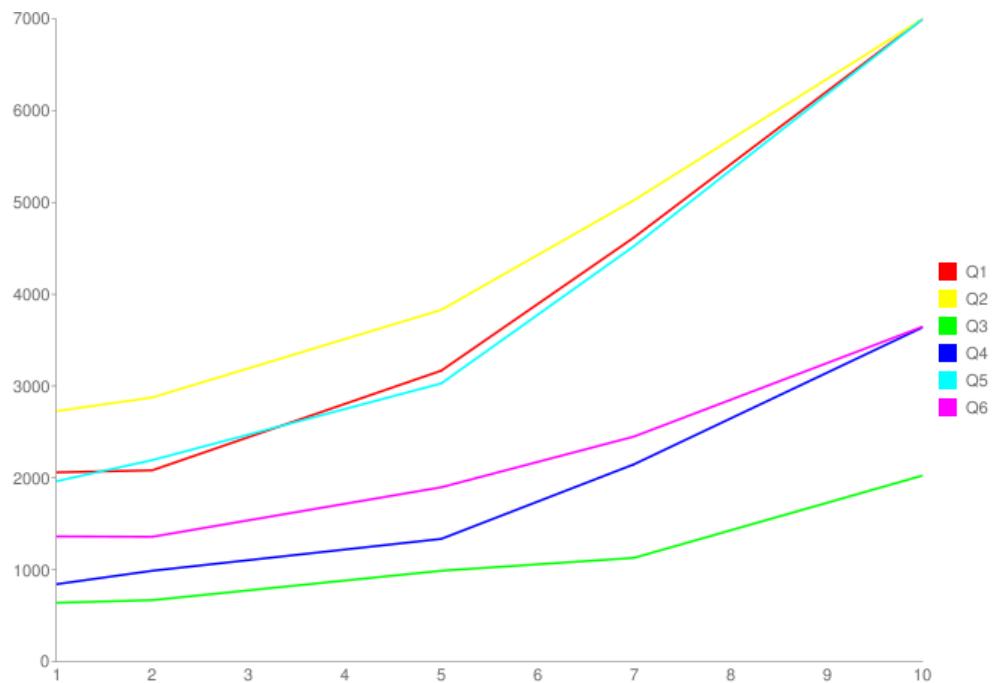


Figure 6.2: Durchschnittliche Response-Time bei einer Wartezeit von 2 Sekunden

Bei noch höherer Server-Belastung benötigen die Queries Q1, Q2 und Q5 bei 10 parallelen Browsern schon durchschnittlich mehr als 7 Sekunden um zu antworten.

6.3.3 Q1 - Q6 mit 4 Sekunden Wartezeit und WITH-Tables

Das Benchmark-Ergebnis mit einer Wartezeit von vier Sekunden unter der Verwendung von WITH-Tables:

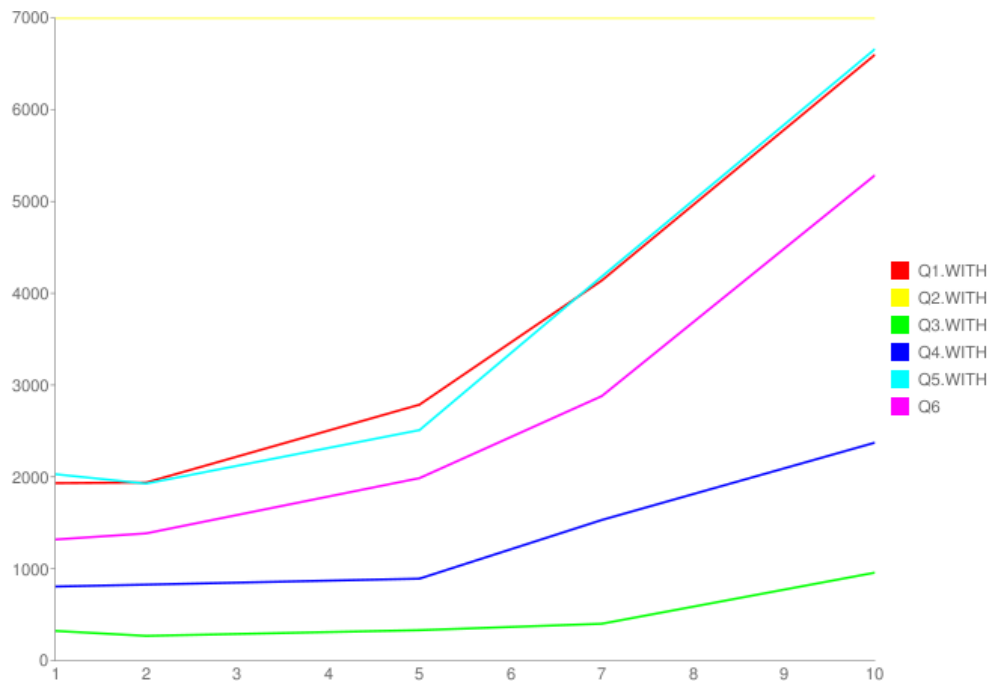


Figure 6.3: Durchschnittliche Response-Time bei einer Wartezeit von 4 Sekunden unter Verwendung von WITH-Tables

Man sieht sofort dass Q2 durch WITH-Tables deutlich langsamer geworden ist. Die anderen Queries sind dadurch im Allgemeinen etwas schneller geworden, insbesondere Q3.

6.3.4 Q1 - Q6 mit 2 Sekunden Wartezeit und WITH-Tables

Das Benchmark-Ergebnis mit einer Wartezeit von zwei Sekunden unter der Verwendung von WITH-Tables:

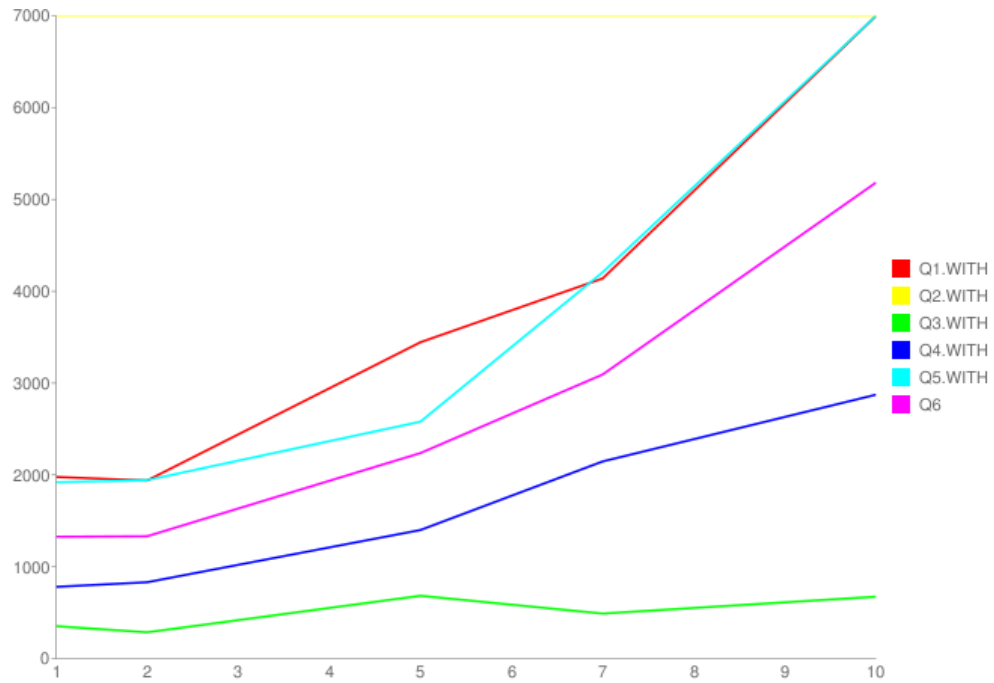


Figure 6.4: Durchschnittliche Response-Time bei einer Wartezeit von 2 Sekunden unter Verwendung von WITH-Tables

Bei noch höherer Server-Belastung skalieren Q3 und Q5 mit WITH-Tables besser als ohne.

6.3.5 Q7 mit 2 Sekunden Wartezeit

Das Benchmark-Ergebnis von Q7 mit einer Wartezeit von zwei Sekunden einmal mit temporären Tabellen und einmal mit `WITH`-Tables:

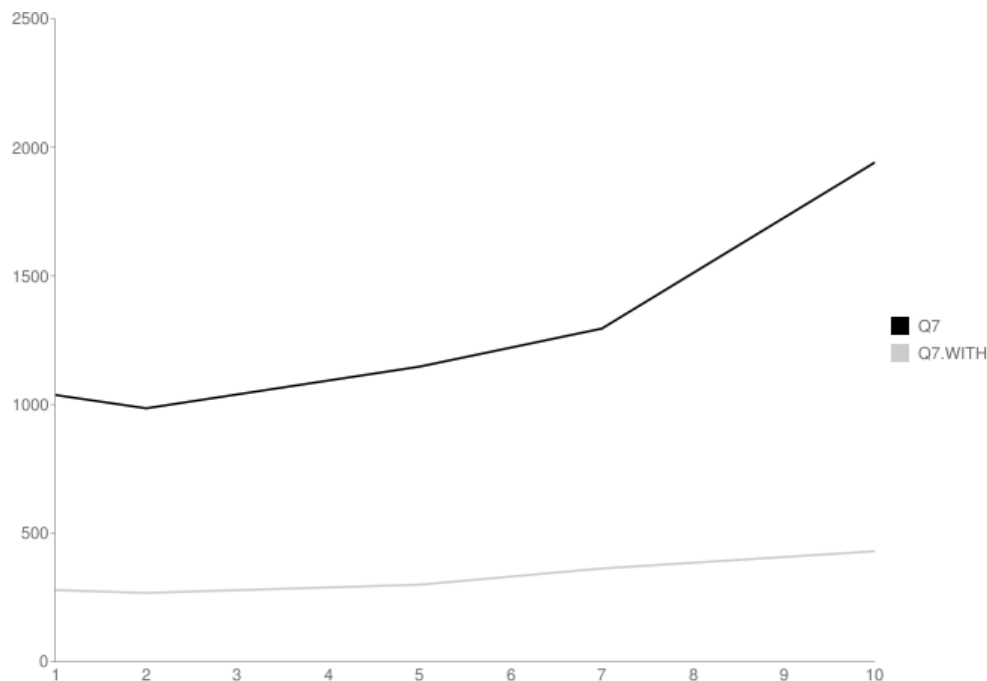


Figure 6.5: Durchschnittliche Response-Time bei einer Wartezeit von 2 Sekunden

Auch Q7 ist mit `WITH`-Tables performanter als ohne.

6.3.6 Interpretation der Ergebnisse

Auf die Laufzeit der Queries wirken sich mehrere Faktoren aus. Wichtige Faktoren sind:

- Effizienz Gestaltung der Queries (z.B. korrelierte Unterabfragen vermeiden)
- Optimierung des Anfragebaumes
- Leistungsfähigkeit der Hardware
- Caching von Ergebnissen durch das DBMS
- Vorhandensein von geeigneten Index-Strukturen

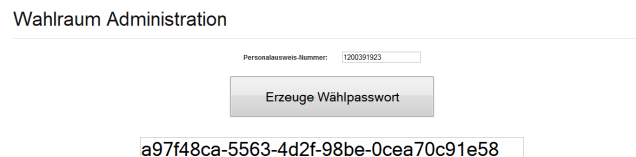
Bei der Durchführung der Queries ist aufgefallen, dass die erste Anfrage oft um Größenordnungen länger für die Ausführung braucht, als die folgenden Abfragen. Der Grund hierfür ist, dass das DBMS Ergebnisse von zeitnah durchgeführten Anfragen oft Speichert um spätere Anfragen schneller durchführen zu können. Im Extremfall wird also das Endergebnis gespeichert und nicht neu berechnet. In der vorliegenden Arbeit würde also

das DBMS am besten abschneiden, dass exakt die Ergebnistabellen der letzten Anfragen zwischenspeichert. Den stärksten Effekt auf die Laufzeit haben Optimierungen, die das DBMS dazu bringen mehr von der Endergebnistabelle zu cachen. Dies ist im vorliegenden Benchmark wohl der Hauptgrund für Performanzunterschiede zwischen SQL-Abfragen basierend auf "WITH"-Befehlen und SQL-Abfragen basierend auf Global Temporary Tables. Der Grund dafür, dass die Anfragen unter Verwendung von "WITH"-Befehlen teilweise auch deutlich langsamer sind, als die Anfragen mit temporary tables ist dementsprechend vermutlich ebenfalls auf andere Caching Eigenschaften zurückzuführen. In einer realen Implementierung sollte der Entwickler das Caching selber übernehmen und bei jeder Browser Anfrage nur die minimal nötigen Informationen neu berechnen.

Zusätzlich hat die Verwendung von "WITH" Befehlen auch den Effekt, dass das DBMS die Anfrage besser optimieren kann, da alle Teiltabellen auf einmal zur Verfügung stehen.

7 Stimmabgabe

Die Stimmabgabe im Wahlinformationssystem läuft folgendermaßen ab. Der Wähler erscheint im Wahllokal seines Wahlbezirkes. Dort legt er seinen Personalausweis vor. Durch die Vorlage des Personalausweises wird verhindert dass eine nicht-autorisierte Person eine Stimme abgeben kann. Der Wahlhelfer vor Ort gibt die Personalausweisnummer in der vorgesehenen Weboberfläche ein:



Wahlraum Administration

Personalausweis-Nummer: 1200391923

Erzeuge Wahlpasswort

a97f48ca-5563-4d2f-98be-0cea70c91e58

Figure 7.1: Webschnittstelle zur Eingabe der Personalausweis-Nummer

Das System prüft ob die Personalausweisnummer ein stimmberechtigter Wähler ist (Ausweisnummer vorhanden). Wenn ja, wird geprüft ob dieser nicht bereits seine Stimme abgegeben hat (Flag *gewählt* muss auf *false* gesetzt sein). Hat er noch nicht gewählt, generiert das System ein zufälliges, zentral auf dem Server gespeichertes Wahlpasswort, das der Wahlhelfer dem Wähler auf einem Zettel übergibt. Durch die Generierung des Wahlpassworts wird das Flag *gewählt* auf *true* gesetzt. Dadurch wird verhindert dass der gleiche Wähler mehrfach wählen kann.

7.1 Wahlzettel

Das Wahlpasswort berechtigt den Wähler an einem Wahlcomputer seine Stimme abzugeben:

Sobald der Wähler seine Kreuze gemacht hat, überprüft das System das eingegebene Wahlpasswort. Wenn es korrekt ist, wird die Stimme gezählt und das Wahlpasswort erlischt. Durch die Indirektion über das anonyme Wahlpasswort wird maximaler Datenschutz gewährleistet.

Alle Übertragungen vom Wahllokal zum Server müssen gesichert sein. Da sichere Verbindungen nicht Thema dieses Projektes sind, wird in der Implementierung davon ausgegangen, dass die Sicherung der Verbindung außerhalb erfolgt.

Stimmzettel für die Wahl zum Deutschen Bundestag im Wahlkreis 12 Wismar - Nordwestmecklenburg - Parchim am 27. September 2009

Sie haben 2 Stimmen



hier 1 Stimme
für die Wahl
eines/einer Wahlkreis-
abgeordneten

Erststimme



hier 1 Stimme
für die Wahl
einer Landesliste (Partei)
- maßgebende Stimme für die Verteilung der Sitze
insgesamt auf die einzelnen Parteien -

Zweitstimme

1	Bliemel, Stephan SPD	<input type="radio"/>
2	Broziat, Martin FDP	<input checked="" type="radio"/>
3	Bunge, Dr. Martina DIE LINKE	<input type="radio"/>
4	Kind, Lutz Parteilos	<input type="radio"/>
5	Köster, Stefan NPD	<input type="radio"/>
6	Seemann-Katz, Ulrike GRÜNE	<input type="radio"/>
7	Strenz, Karin CDU	<input type="radio"/>
8		
9		
10		

<input type="radio"/>	SPD Sozialdemokratische Partei Deutschlands	1
<input type="radio"/>	FDP Freie Demokratische Partei	2
<input checked="" type="radio"/>	DIE LINKE DIE LINKE	3
		4
<input type="radio"/>	NPD Nationaldemokratische Partei Deutschlands	5
<input type="radio"/>	GRÜNE BÜNDNIS 90/DIE GRÜNEN	6
<input type="radio"/>	CDU Christlich Demokratische Union Deutschlands	7
<input type="radio"/>	REP DIE REPUBLIKANER	8
<input type="radio"/>	MLPD Marxistisch-Leninistische Partei Deutschlands	9
<input type="radio"/>	PIRATEN Piratenpartei Deutschland	10

Wählpasswort: a97f48ca-5563-4d2f-98be-0cea70c91e58

Sobald Sie klicken gilt Ihre Stimme als abgegeben!

Wählen

Figure 7.2: Digitaler Wahlzettel mit Wählpasswort

7.2 Auswertung

Durch einen bestimmten Befehl im Webinterface kann die Auszählung der Stimmen auf Wahlkreisebene angestoßen werden. Es wird davon ausgegangen, dass der Befehl nur dem Organisator bekannt ist und nur von physisch geschützten Orten aus gegeben werden kann. In unserer Implementierung wird die Auszählung durch <http://localhost:8080/WahlWebsite/ShowResult?query=refreshvotes> ausgelöst. Der Begriff `refreshvotes` steht dabei als Beispiel für einen möglichen geheimen Befehl. Tatsächlich sollte dieser Befehl einem langen Passwort entsprechen. Die Auszählung darf erst nach Schließung des letzten Wahllokales ausgelöst werden. Die Sicherung der Auswertungsfunktion ist nötig, um zu verhindern, dass ständig hochaktuelle Ergebnisse verfügbar sind. Hochaktuelle Ergebnisse würden es in Einzelfällen möglich machen, auf die Stimmabgabe von Einzelpersonen zu schließen, indem das Ergebnis vor der Stimmabgabe mit dem Ergebnis nach der Stimmabgabe verglichen wird.

Bibliography