Logo, company name

Description automatically generated

Vor 2021

T-202-GAG1, Gagnasafnsfræði

Hópaverkefni 5

Ástþór Arnórsson, astthor19@ru.is

Ingólfur Orri Gústafsson, ingolfurg19@ru.is

Viktoría Inga Smáradóttir, viktoria19@ru.is 091299-2919

31.mars, 2021

Kennari: Anna Sigríður Islind

**1. Introduction**

This is a report for the assignment project 5 in Gagnasafnsfræði. In this assignment we are finding functional- and multi-valued dependencies. We then find the minimal cover of these dependencies and the normal form of the relation. Lastly we decompose the relations and find their dependencies and normal form. We then create the relations in PostgreSQL in the file DECOMPOSITION.sql and we fill these relations in POPULATE.sql.

**2. Finding FDs and MVDs**

We wrote two simple queries to find all functional dependencies and multi-valued dependencies in the original relations. We wrote a single generator.py file which generates either FDs and MVDs depending on if the user inputs “fd” or “mvd”. The generator then writes to the file prj5\_check\_FD.sql for the FDs and prj5\_check\_MVD.sql for the MVDs. To find the dependencies we then run the sql files as queries.

**2.1 Finding FDs**

This query checks for dependencies A → B. It can happen that the queries finds FDs that go both ways, that is A → B and B → A where A is the key. The B → A FDs are ignored.

FD\_template = '''

    SELECT '{R}: {A} --> {B}' AS FD,

    CASE WHEN COUNT(\*) = 0 THEN

        'HOLDS'

    ELSE

        'does not hold'

    END AS VALIDITY

    FROM (

        SELECT {A}

        FROM {R}

        GROUP BY {A}

        HAVING COUNT(DISTINCT {B}) > 1

    ) X;

'''

**2.2 Finding MVDs**

This query checks for dependencies A → B,C. We however only consider MVDs for relations that have a primary key combined of three or more columns.

MVD\_template = '''

    SELECT

        '{A} ->> ({B}, {C}) in {R}' AS Relation,

        CASE WHEN COUNT(\*) = 0 THEN

            'MAYBE MVD'

        ELSE

            'NO MVD'

        END AS MVD

    FROM (

        SELECT {A}

        FROM {R}

        GROUP BY {A}

        HAVING COUNT(\*) > 1

            AND COUNT(\*) <> COUNT(DISTINCT {B}) \* COUNT(DISTINCT {C})

    ) X;

'''

**3. Normalization**

**3.1 Normalization of CivilServices**

Primary key: CSID, HID

Determined FDs:

CSID → PN

HID → HS

HID → HZ

HID  → HC

HZ → HC

(CSID, HID) → PN

(CSID, HID) → S

(CSID, HID) → HS

(CSID, HID) → HZ

(CSID, HID) → HC

Minimal cover:

CSID → PN

HID → HS

HID → HZ

HZ → HC

(CSID, HID) → S

Other keys: None

Normal form: 1NF because a key → non key.

Decomposition:

CivilServices\_CSID\_PN

CivilServices\_HID\_HS\_\_HZ

CivilServices\_HZ\_HC

CivilServices\_CSID\_HID\_S

**The new relations**

**CivilServices\_CSID\_PN**

Columns: CSID, PN

Key: CSID

FDs: CSID → PN

Normal forms:

* Since all FDs are key FDs, the table is in BCNF
* Since the key has a single column, the table is in 4NF

**CivilServices\_HID\_HS\_\_HZ**

Columns: HID, HS, HZ

Key: HID

FDs: HID → HS, HID → HZ

Normal forms:

* Since all FDs are key FDs, the table is in BCNF
* Since the key has a single column, the table is in 4NF

**CivilServices\_CSID\_HID\_S**

Columns: CSID, HID, S

Key: CSID, HID

FDs: (CSID, HID) → S

Normal forms:

* Since all FDs are key FDs, the table is in BCNF
* Since the key is only two columns, MVDs were not considered and the table should be in 4NF

**CivilServices\_HZ\_HC**

Columns: HZ, HC

Key: HZ

FDs: HZ → HC

Normal forms:

* Since all FDs are key FDs, the table is in BCNF
* Since the key has a single column, the table is in 4NF

**3.2 Normalization of Projects**

Primary keys: ID, PID, SID

Determined FDs:

ID → MID

PID → PN

SID → SN

MID → MN

(ID, PID, SID) → PN

(ID, PID, SID) → SN

(ID, PID, SID) → MID

MVDs: There were no MVDs for projects.

Other keys: None

Minimal Cover:

ID → MID

PID → PN

SID → SN

MID → MN

Normal forms: 1.NF because some FDs’ are key → non-key.

Decomposition:

Projects\_PID\_PN(\_PID\_, PN)

Projects\_SID\_SN(\_SID\_, SN)

Projects\_ID\_MID(\_ID\_, MID)

Projects\_MID\_MN(\_MID\_, MN)

**The new relations**

**Projects\_PID\_PN**

Columns: PID, PN

Keys: PID

FDs: PID → SN

Normal form:

* The table is in BCNF because all FDs are key FDs
* Because the key has a single column the table is in 4NF

**Projects\_SID\_SN**

Columns: SID, SN

Keys: SID

FDs: SID → SN

Normal form:

* The table is in BCNF because all FDs are key FDs
* Because the key has a single column the table is in 4NF

**Projects\_ID\_MID**

Columns: ID, MID

Keys: ID

FDs: ID → MID

Normal form:

* The table is in BCNF because all FDs are key FDs
* Because the key has a single column the table is in 4NF

**Projects\_MID\_MN**

Columns: MID, MN

Keys: MID

FD: MID → MN

Normal form:

* The table is in BCNF because all FDs are key FDs
* Because the key has a single column the table is in 4NF

**3.3 Normalization of Citizens**

Primary key: CID

Determined FDs:

CID → CN

CID → CS

CID → CNr

CID → CZ

CID → CL

CID → EID

CZ → CL

Minimal cover:

CID → CN

CID → CS

CID → CNr

CID → CZ

CID → EID

CZ → CL

Other keys: None

Normal form:  2NF because in one FD a non-key → non-key.

Decomposition:

Citizens\_CID\_CN\_CS\_CNr\_CZ\_EID

Citizen\_CZ\_CL

**The new relations**

**Citizens\_CID\_CN\_CS\_CNr\_CZ\_EID**

Columns: CID, CN, CS, CNr, CZ, EID

Key: CID

FDs: CID → CN, CS, CNr, CZ, EID

Normal forms:

* This relation is BCNF, since all FDs are key FDs
* Since the key only has one column, the table is in 4.NF

**Citizens\_CZ\_CL**

Columns: CZ, CL

Key: CZ

FDs: CZ → CL

Normal forms:

* This relation is BCNF, since all FDs are key FDs
* Since the key only has one column, the table is in 4.NF

**3.4 Normalization of Coffees**

Primary keys: DID CID HID

Determined FDs:

DID → DN

DID → DS

CID → CN

CID → CC

(DID, HID, CID) → DN

(DID, HID, CID) → DS

(DID, HID, CID) → DN

(DID, HID, CID) → CN

(DID, HID, CID) → CC

MVDs: DID → HID, CID

Other keys: None

Minimal cover:

DID → DN

DID → DS

CID → CN

CID → CC

DID CID HID → DID CID HID

Normal form: 1.NF since FDs are key → non-key.

Decomposition:

Coffees\_DID\_DN\_DS(\_DID\_, DN, DS)

Coffees\_CID\_CN\_CC(\_CID\_, CN, CC)

Coffees\_DID\_HID\_CID(\_DID\_HID\_CID\_)

**The new relations**

Coffees\_DID\_DN\_DS

Columns: DID, DN, DS

Keys: DID

FDs: DID → DN, DID → DS

Normal form:

* This relation is BCNF, since all FDs are key FDs
* Since the key only has one column, the table is in 4NF

**Coffees\_CID\_CN\_CC**

Columns: CID, CN, CC

Keys: CID

FDs: CID → CN, CID → CC

Normal form:

* This relation is BCNF, since all FDs are key FDs
* Since the key only has one column, the table is in 4NF

**Coffees\_DID\_HID\_CID**

Columns: DID, HID, CID

Keys: DID, HID, CID

FDs: None

Normal form:

* Since each attribute is a key the relation is in BCNF.
* We looked at the MVDs for the coffees table which gave: DID → → HID, CID. Now we know that HID and CID are both derived from DID. The table is therefore not in 4NF and it must be decomposed further.

Decomposition: Coffees\_DID\_HID and Coffees\_DID\_CID

**Coffees\_DID\_HID**

Columns: DID, HID

Keys: DID, HID

FDs: None

Normal form:

* Since the attributes are only keys it is in BCNF
* The table has only two attributes so it is so small for MVDs and the table is in 4NF

**Coffees\_DID\_CID**

Columns: DID, CID

Keys: DID, CID

FDs: None

Normal form:

* Since the attributes are only keys it is in BCNF
* The table has only two attributes so it is so small for MVDs and the table is in 4NF