Data Management

Coursework 2

Berke Galadari

bg4u17 — 29543371

1 The Relational Model

1.1 EX1

Relation
(varchar: dateRep, int: day, int: month, int: year, int: cases, int: deaths, varchar: countriesAndTerritories, varchar: geoId, varchar: countryterritoryCode, varchar: popData2018, varchar: continentExp)

1.2 EX2

countriesAndTerritories $\rightarrow$ $geoId, countryterritoryCode, popData2018, continentExp$
geoId $\rightarrow$ $countriesAndTerritories, countryterritoryCode, popData2018, continentExp$
dateRep $\rightarrow$ $day, month, year$
dateRep, countriesAndTerritories, geoId $\rightarrow$ $day, month, year, cases, deaths, countryterritoryCode, popData2018, continentExp$

Assumed that the attributes "countryterritoryCode" and "popData2018" may take a null value, or may not be accurate in determining unique elements.

1.3 EX3

(dateRep, geoId)
(dateRep, countriesAndTerritories)

(day, month, year) can be used instead of (dateRep) in all of the above. Even (day, month) can be used for the time being but would be unusable once one year anniversary of COVID-19 comes.

1.4 EX4

dateRep, geoId $\rightarrow day, month, year, cases, deaths,$
$countriesAndTerritories, countryTerritoryCode, popData2018, continentExp$

2 Normalisation

2.1 EX5 AND EX6

Partial-Key Dependencies

dateRep $\rightarrow day, month, year$
countriesAndTerritories $\rightarrow countryterritoryCode, popData2018, continentExp$
geoId $\rightarrow countryterritoryCode, popData2018, continentExp$

First, decompose original table into two separate tables.
1.(varchar: dateRep, int: day, int: month, int: year, varchar: geoId,
int: cases, int: deaths)

2.(varchar: geoId, varchar: countriesAndTerritories,
varchar: countryterritoryCode, varchar: popData2018,
varchar: continentExp)

Then decompose (1.) further into:
3.(varchar: dateRep, varchar: geoId, int: day, int: month, int: year)
4.(varchar: dateRep, varchar: geoId, int: cases, int: deaths)

2.2 EX7 AND EX8

Transitive Dependencies

geoId $\rightarrow$ **countriesAndTerritories** $\rightarrow$ $countryterritoryCode, popData2018, continentExp$
countriesAndTerritories $\rightarrow$ **geoId** $\rightarrow$ $countryterritoryCode, popData2018, continentExp$

Decompose table(2.) from EX5 and EX6 into:
5.(varchar: geoId, varchar: countriesAndTerritories)
6.(varchar: countriesAndTerritories, varchar: countryterritoryCode, varchar: popData2018, varchar: continentExp)

2.3 EX9

Yes, this relation is in BCNF. This is because there is only one candidate key in relation. TEKRAR KONTROL ET

3 Modelling

3.1 EX10

The csv dataset was imported into a table called dataset in coronavirus.db.

3.2 EX11

```
-- Table: R1
CREATE TABLE R1(
    dateRep VARCHAR,
    geoId VARCHAR,
    day INT,
    month INT,
    year INT,
```

```
    PRIMARY KEY (dateRep, geoId),
    FOREIGN KEY (dateRep, geoId)
        REFERENCES R2 (dateRep, geoId)
);

-- Table: R2
CREATE TABLE R2(
    dateRep VARCHAR,
    geoId VARCHAR,
    cases INT,
    deaths INT,
    PRIMARY KEY (dateRep, geoId),
    FOREIGN KEY (geoId)
        REFERENCES R3 (geoId)
);

-- Table: R3
CREATE TABLE R3(
    geoId VARCHAR,
    countriesAndTerritories VARHCAR,
    PRIMARY KEY (geoId),
    FOREIGN KEY (countriesAndTerritories)
        REFERENCES R4(countriesAndTerritories)
);

-- Table: R4
CREATE TABLE R4(
    countriesAndTerritories VARCHAR,
    countryterritoryCode VARCHAR,
    popData2018 VARCHAR,
    continentExp VARCHAR,
    PRIMARY KEY (countriesAndTerritories)
);
```

3.3 EX12

```
INSERT INTO R1 (dateRep,geoId,day,month,year)
SELECT DISTINCT dateRep,geoId,day,month,year FROM dataset;

INSERT INTO R2 (dateRep,geoId,cases,deaths)
SELECT DISTINCT dateRep,geoId,cases,deaths FROM dataset;

INSERT INTO R3 (geoId,countriesAndTerritories)
SELECT DISTINCT geoId,countriesAndTerritories FROM dataset;

INSERT INTO R4 (countriesAndTerritories,countryterritoryCode,
popData2018,continentExp)
SELECT DISTINCT countriesAndTerritories,countryterritoryCode,
popData2018,continentExp FROM dataset;
```

3.4 EX13

Tested on clean SQLite database, database was populated successfully.

4 Querying

4.1 EX14

```
SELECT SUM(cases),SUM(deaths) FROM R2;
```

4.2 EX15

```
SELECT dateRep,cases FROM dataset
WHERE geoId = 'UK'
ORDER BY year,month,dateRep;
```

4.3 EX16

```
SELECT dateRep,continentExp,SUM(cases) totalCases,
SUM(deaths) totalDeaths FROM dataset
GROUP BY dateRep,continentExp
ORDER BY continentExp,year,month,dateRep;
```

4.4 EX17

```
SELECT countriesAndTerritories,
SUM(cases)*100/popData2018 populationPercentageCases,
SUM(deaths)*100/popData2018 populationPercentageDeaths
FROM dataset
GROUP BY countriesAndTerritories;
```

4.5 EX18

```
SELECT countriesAndTerritories,
SUM(deaths)*100/SUM(cases) deathPercentageForTotalCases
FROM dataset
GROUP BY countriesAndTerritories
ORDER BY deathPercentageForTotalCases DESC LIMIT 10;
```

4.6 EX19

```
SELECT dateRep,SUM(cases) OVER
(ORDER BY year,month,dateRep
ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW),
SUM(deaths) OVER
(ORDER BY year,month,dateRep
ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW)
FROM dataset
WHERE geoId = 'UK'
GROUP BY dateRep
ORDER BY year,month,dateRep;
```