Tesina Basi Di Dati

Pasquale del Moro, Dario Di Meo

**Indice**

1. Introduzione
2. Normalizzazione Master Table Province
3. Creazione Basi Di Dati
4. Arrichimento base dati
5. Modello E/R
6. Analisi dei dati
   1. Query
   2. Grafici
7. Stored Procedure
8. Ottimizzazione Base di dati

**Appendice**

1. Creazione Master table
2. Fonti e siti web utili
3. **INTRODUZIONE**

La tesina consiste in un progetto di basi di dati, relativo al **COVID-19**, nel contesto dello Stato Italiano e in un arco temporale che va dal **24/02/2020** al **03/05/2020**.

Tale progetto parte da dati relativi alle regioni e alle province italiane forniti dalla protezione Civile Italiana, e ha come idea generale quella di creare una base di dati attiva con la quale poter poi interagire, per analizzare dati ed eventualmente costruire grafici ed infine visualizzarli.

Dalla precedente descrizione è quindi possibile individuare i punti in cui sequenzialmente si è sviluppato il progetto:

* Conversione dei dati forniti dalla protezione civile in tabelle del **modello relazionale**;
* **Normalizzazione** di tali tabelle al fine di ricavare lo schema principale della base dati;
* Aggiunta di ulteriori dati prelevati da altre fonti;
* Estrazione di un possibile **modello ER** corrispondente allo schema di basi di dati ottenuto;
* Scelta di un **insieme di query** da effettuare sul database per la selezione e l’estrazione di dati significativi;
* Realizzazione di **trigger e stored procedure** che permettono rispettivamente di implementare politiche di reazione e parametrizzare query;
* Creazione di **grafici** sulla base dei risultati delle query e relativi alle grandezze significative della base di dati;
* Creazione di uno strumento di visualizzazione grafica, nello specifico un **istogramma animato**, realizzato nel linguaggio Python e con la libreria grafica pygame;
* Definizione di indici e viste al fine di migliorare le prestazioni di alcune query ricorrenti.

Da tale elenco si deduce la necessità di normalizzare le tabelle estratte dai dati della protezione civile, per poter poi lavorare agilmente sulla **base di dati** e renderla **attiva** con trigger e stored produre.

Alla creazione della base di dati e alla loro analisi, si è aggiunta anche una parte del progetto dedicata alla definizione, tramite un processo di **Reverse Engineering**, dello **schema concettuale** E**/**R, per rappresentare la realtà di nostro interesse.

La realizzazione fisica della base di dati è stata effettuata mediante il DBMS locale Oracle 18c XE insieme all’ambiente di sviluppo SQL-DEVELOPER.

1. **Normalizzazione Master Table Province**

Estratta la tabella dal file **Csv** riguardante il **COVID-19** e relativa alle **province italiane** dal **24/02/2020** al **03/05/2020**, si vuole rendere in **3a forma normale** tale tabella.

**-----------------------------------------------------------------------------------**

Lo **schema della tabella estratta** è il seguente:

**MASTER\_TABLE\_PROVINCE**

**( data**, stato, codice\_regione, denominazione\_regione, **codice­\_provincia**, denominazione\_provincia, sigla\_provincia, latitudine, longitudine, totale\_casi, note\_in, note\_en **)**

dove la **chiave primaria** è costituita dalla coppia di attributi *(****data, codice\_provincia****)*;

inoltre i tipi corrispondenti ai campi sono stringhe di testo o numerici, tranne per la data, che è di tipo DATE.

-----------------------------------------------------------------------------------

**PRIMA FORMA NORMALE**

La tabella estratta risulta essere in 1a forma normale in quanto banalmente il suo schema contiene solo attributi semplici, quindi né strutturati né multi-valore.

**INDIVIDUAZIONE DELE DIPENDENZE FUNZIONALI NON BANALI**

**{ codice\_provincia, data }  {** tutti gli altri attributi **}**

Essendo {codice\_provincia, data} la chiave primaria

**{ codice\_provincia } **stato, codice\_regione, denominazione\_regione, denominazione\_provincia, sigla\_provincia, latitudine, longitudine

**{ codice\_regione } **stato, denominazione\_regione

**SECONDA FORMA NORMALE**

Ricerca delle eventuali dipendenze funzionali non complete dalla chiave agli attributi non primi,

ossia ricerca di dipendenze funzionali da una sotto-parte della chiave ad un altro attributo.

Attributi primi che dipendono esclusivamente da ***codice\_provincia***:

***codice\_provincia****denominazione\_provincia*, *sigla\_provincia*, *latitudine, longitudine*; *codice\_regione*,*denominazione\_regione, stato*.

**Lo schema di relazione non risulta essere in 2a normale**.

Schemi in 2a forma normale:

**MISURE\_PROVINCE**( **data**,**codice­\_provincia**:PROVINCE, totale\_casi, note\_in, note\_en )

**PROVINCE**( stato, codice\_regione, denominazione\_regione, **codice­\_provincia**, denominazione\_provincia, sigla\_provincia, latitudine, longitudine )

**TERZA FORMA NORMALE**

Eliminazione delle eventuali dipendenze funzionali transitive tra chiave ed attributi non primi,

ossia individuazione di tutti gli attributi non primi che dipendono da altri attributi non primi.

* **Tabella dei contagi MISURE\_PROVINCE**

Per tale schema si verifica che fra i tre attributi non primi *totale\_casi*, *note\_in*,*note\_en* non vi sono dipendenze funzionali, quindi non sussistono dipendenze transitive dalla chiave, il che implica che **la 3a forma normale è già verificata**.

* **Tabella dei dati relativi alle *PROVINCE***

Il codice di una regione la identifica univocamente, per cui tutti gli attributi relativi a informazioni della regione sono in dipendenza **solo** da *codice\_regione*: gli attributi *stato* e *denominazione\_regione* dipendono dall’attributo *codice\_regione*. Quindi si ha una dipendenza transitiva da ***codice\_provincia*** a *(stato, denominazione\_regione).*

**Lo schema di relazione non risulta essere in 3a normale**.

Schema in 3a forma normale:

**PROVINCE**( codice\_regione:REGIONI, **codice­\_provincia**, denominazione\_provincia, sigla\_provincia, latitudine, longitudine ).

**REGIONI**( stato, **codice\_regione**, denominazione\_regione )

**RISULTATO DELLA NORMALIZZAZIONE**

Riordinando gli attributi in maniera più opportuna, si hanno i seguenti schemi:

**-----------------------------------------------------------------------------------**

**MISURE\_PROVINCE**

( **data**, **codice­\_provincia**:PROVINCE, totale\_casi, note\_in, note\_en )

**PROVINCE**

( **codice­\_provincia**, denominazione\_provincia, sigla\_provincia, codice\_regione:REGIONI, latitudine, longitudine )

**REGIONI**

(**codice\_regione**, denominazione\_regione, stato )

**-----------------------------------------------------------------------------------**

1. **Creazione Basi Di Dati**

La creazione della base dati, consiste nella realizzazione fisica del DB. Principalmente si sviluppa in:

* + Definizione di utenti
  + Creazione tabelle generate dal processo di normalizzazione delle Master Table
  + Popolazione delle tabelle

**Definizione di utenti**

**CREATE USER**covid\_dba**IDENTIFIED BY** \*\*\*\*\*\*\*\*\*;

**GRANT UNLIMITEDTABLESPACETO**covid\_dba;

**-----------------------------------------------------------------------------------**

L’utente COVID\_DBA, identificato da una password, ricopre il ruolo di Data Base Administrator, a cui, grazie all’istruzione **GRANT UNLIMITED TABLESPACE,** si rende accessibile tutto lo spazio disponibile. In questo modo è libero di creare un numero indefinito di tabelle.

**Creazione tabelle generate dal processo di normalizzazione delle Master Table**

In seguito alla creazione delle MASTER TABLE (appendice A) e dello schema normalizzato, possiamo creare le tabelle sul nostro DATABASE.

**CREATE TABLE**regioni(

codice\_regione **NUMBER(2),**

denominazione\_regione **VARCHAR2(200),**

Stato **CHAR(3)**

);

**ALTER TABLE**regioni **ADD CONSTRAINT PK\_REGIONI**primary key (codice\_regione);

**CREATE TABLE** province(

codice\_provincia **NUMBER(3),**

denominazione\_provincia **VARCHAR2(200),**

sigla\_provincia **CHAR(2)**,

codice\_regione **NUMBER(2),**

latitudine **NUMBER,**

longitudine **NUMBER**

);

**ALTER TABLE**province**ADD CONSTRAINTPK\_PROVINCE**primary key (codice\_provincia);

**ALTER TABLE** province**ADD CONSTRAINTFK\_PROVINCE\_REGIONI**foreign key (codice\_regione) **REFERENCES** regioni(codice\_regione);

**CREATE TABLE**misure\_province(

data **DATE**,

codice\_provincia **NUMBER(3),**

totale\_casi **INTEGER,**

note\_it **CLOB**,

note\_en **CLOB**

);

**ALTER TABLE**misure\_province**ADD CONSTRAINT PK\_MISURE\_PROVINCE**primary key (data, codice\_provincia);

**ALTER TABLE**misure\_province**ADD CONSTRAINTFK\_MISURE\_PROVINCIA\_PROVINCIA**foreign key (codice\_provincia) **REFERENCES**province(codice\_provincia);

**CREATE TABLE** misure\_regioni(

data **DATE**,

codice\_regione **NUMBER(2),**

ricoverati\_con\_sintomi **INTEGER**,

terapia\_intensiva **INTEGER,**

totale\_ospedalizzati **INTEGER,**

isolamento\_domiciliare **INTEGER,**

totale\_positivi **INTEGER,**

variazione\_totale\_positivi **INTEGER,**

nuovi\_positivi **INTEGER,**

dimessi\_guariti **INTEGER,**

deceduti **INTEGER,**

totale\_casi **INTEGER,**

tamponi **INTEGER,**

casi\_testati **INTEGER,**

note\_it **CLOB,**

note\_en **CLOB**

);

**ALTER TABLE**misure\_regioni**ADD CONSTRAINT PK\_MISURE\_REGIONI** primary key (data, codice\_regione);

**ALTER TABLE**misure\_regioni**ADD CONSTRAINT FK\_MISURE\_REGIONI\_REGIONI** foreign key (codice\_regione) **REFERENCES** regioni(codice\_regione);

**Popolazione delle tabelle**

Le Master Table, oltre ad essere un supporto momentaneo, rendono molto semplici i popolamenti delle tabelle, mediante INSERT con query innestate.

**INSERT INTO PROVINCE**(codice\_provincia, denominazione\_provincia, sigla\_provincia, codice\_regione, latitudine, longitudine) (

**SELECT DISTINCT** codice\_provincia, denominazione\_provincia, sigla\_provincia, codice\_regione,

latitudine, longitudine

**FROM**master\_table\_province

);

**INSERT INTO**regioni(codice\_regione, denominazione\_regione, stato) (

**SELECT DISTINCT** codice\_regione, denominazione\_regione, stato

**FROM** master\_table\_province

);

**INSERT INTO**misure\_province (codice\_provincia, data, totale\_casi, note\_en, note\_it)(

**SELECT** codice\_provincia, data, totale\_casi, note\_en, note\_it

**FROM** master\_table\_province

);

**INSERT INTO**misure\_regioni(

**SELECT**data,codice\_regione,ricoverati\_con\_sintomi,terapia\_intensiva,totale\_ospedalizzati,isolamento\_domiciliare,totale\_positivi,variazione\_totale\_positivi,nuovi\_positivi,dimessi\_guariti,deceduti,totale\_casi,tamponi,casi\_testati,note\_it,note\_en

**FROM**master\_table\_regioni

);

1. **Arrichimento base dati**

**ESTENSIONE DELLE TABELLE PRESENTI**

Ottenute le tabelle sul DB, si è scelto di aggiungere ulteriori dati prelevati da altre fonti, nello specifico:

Relativamente alla tabella PROVINCE:

* la popolazione in numero di abitanti, popolazione,
* l’estensione territoriale in Km quadrati, superficie\_kmq,
* la densità abitativa, densita\_abitativa,

Relativamente alla tabella REGIONI:

* il numero totale di posti letto, posti\_letto,
* il numero totale di posti di terapia intensiva,posti\_terapia\_intensiva.

Comandi SQL per l’alterazione degli schemi delle tabelle:

Per PROVINCE:

**ALTER TABLE** province **ADD** popolazione **INTEGER**;

**ALTER TABLE** province **ADD** superficie\_kmq **INTEGER**;

**ALTER TABLE** province **ADD** densita\_abitativa **INTEGER**;

Per REGIONI:

**ALTER TABLE** regioni **ADD** posti\_letto **INTEGER**;

**ALTER TABLE** regioni **ADD** posti\_terapia\_intensiva **INTEGER**;

**AGGIUNTA DI UNA NUOVA TABELLA**

Un’ulteriore operazione di arricchimento della base di dati è stata quella di creare una nuova tabella relativa alle regioni, MOVIMENTI\_REGIONI, contenente informazioni sulla variazione percentuale della media di alcune grandezze relative agli spostamenti regionali.

Comando SQL per la creazione della tabella MOVIMENTI\_REGIONI:

**CREATE TABLE**movimenti\_regioni(

codice\_regione **NUMBER(2),**

data **DATE**,

variazione\_svago\_vendita\_dettaglio **NUMBER(3),**

variazione\_farmacie\_Alimentari **NUMBER(3),**

variazione\_parchi **NUMBER(3),**

variazione\_stazioni\_trasporto\_pubblico **NUMBER(3),**

variazione\_sedi\_lavoro **NUMBER(3),**

variazione\_zone\_residenziali **NUMBER(3)**

);

Una volta creata la tabella, va considerata la chiave primaria, che è { codice\_regione, data }, e il fatto che ogni tupla deve riferirsi ad una regione, pertanto { codice\_regione } deve essere chiave esterna referenziante la tabella REGIONI.

Comandi SQL per la definizione delle chiave primaria e della chiave esterna:

**ALTER TABLE** movimenti\_regioni **ADD CONSTRAINT PK\_MOVIMENTI\_REGIONI** primary key (data, codice\_regione);

**ALTER TABLE** movimenti\_regioni **ADD CONSTRAINT FK\_MOVIMENTI\_REGIONE\_REGIONE** foreign key(codice\_regione) **REFERENCES** regioni(codice\_regione);

1. **Modello E/R**

PARCHI

LAVORO

FARMACIE

VARIAZIONE\_SPOSTAMENTO

SVAGO

DATA

ETC…

(1,1)

**LUOGO\_**

**MISURAZIONE**

**3**

**MISURA MOVIMENTI**

CODICE\_REGIONE

DENOMINAZIONE\_PROVINCIA

CODICE\_PROVINCIA

DENOMINAZIONE\_REGIONE

CODICE\_REGIONE

(0,N)

ETC…

TAMPONI

DECEDUTI

TERAPIA\_INTENSIVA

TOTALE\_CASI

(0,N)

DATA

LONGITUDINE

LATITUDINE

POSIZIONE

(0,N)

**LUOGO\_**

**MISURAZIONE**

**2**

CODICE\_PROVINCIA

CODICE\_REGIONE

(1,1)

EN

ITA

NOTE

(1,1)

**SITO**

**MISURA PROVINCIA**

**MISURA REGIONE**

**MISURA COVID**

**REGIONE**

**PROVINCIA**

**LUOGO\_**

**MISURAZIONE**

**1**

(1,1)

(1,N)

ETC…

ETC…

DATA

**INDIVIDUAZIONE DELLE ENTITA’**

**REGIONE** e **PROVINCIA** costituiscono delle entità;

anche per le misure ad esse relative si può pensare a delle entità: in particolare è possibile individuare una gerarchia relativa alle misurazioni sul covid-19 costituita da un’entità madre avente attributi generali (ad esempio totale\_casi) e due entità figlie, una per le misurazioni provinciali, una per quelle regionali;dunque l’entità madre sarà **MISURA\_COVID**, mentre le entità da essa specializzate saranno **MISURA\_REGIONE** e **MISURA\_PROVINCIA**.

Infine, dallo schema **MOVIMENTI\_REGIONI**è possibile ricavare un’ulteriore entità **MISURA\_MOVIMENTI**.

**INDIVIDUAZIONE DELLE ASSOCIAZIONI**

Le entità **REGIONE** e **PROVINCIA** sono in associazione **uno a molti**, con partecipazione **non opzionale**, in quanto una provincia è situata in una ed una sola regione, mentre in una regione possono essere situate una o più province, tale associazione è **Sito**.

Le entità **REGIONE** e **PROVINCIA** sono in associazione rispettivamente con **MISURA\_REGIONE** e **MISURA\_PROVINCIA**; entrambe le associazioni **Luogo\_Misurazione\_1** e **Luogo\_Misurazione\_2**  sono di tipo **uno a molti**, con cardinalità minima pari a 0 dal lato **REGIONE** e **PROVINCIA** e cardinalità minima pari a 1 verso **MISURA\_REGIONE**e **MISURA\_PROVINCIA**, in quanto una certa misurazione deve necessariamente essere associata ad una e una sola località, mentre ad una data località potrebbe non essere associata alcuna misurazione o più di una.

Inoltre entrambe le associazioni hanno un attributo proprio che è la **data**.

Un’importante osservazione su queste due associazioni è che le entità **MISURA\_REGIONE** e **MISURA\_PROVINCE** della gerarchia **MISURA\_COVID** sono prive di identificatori interni, mentre hanno due identificatori esterni: uno è il **codice della regione** o della **provincia** a cui si riferiscono, l’altro la **data**; **dunque anche l’esistenza dell’identificatore esterno obbliga che la cardinalità minima sia pari a 1, e non 0, dal lato delle misurazioni**.

Infine si ha un’associazione tra le entità **MISURA\_MOVIMENTI** e **REGIONE**, che è **Luogo\_Misurazione\_3** ed è di tipo **uno a molti**, sempre con cardinalità minima pari a 0 dal lato **REGIONE** e pari a 1 dall’altro lato; sempre con un attributo proprio che è la data, e quest’ultima, insieme all’identificatore di **REGIONE**, fa da identificatore esterno di **MISURA\_MOVIMENTI.**

**INDIVIDUAZIONE DEGLI ATTRIBUTI**

L’entità **REGIONE** è costituita dai soli attributi semplici codice, denominazione, stato,

dove codice fa da identificatore.

L’entità **PROVINCIA** è costituita dagli attributi semplici codice, denominazione, sigla, dove codice fa da identificatore, e poi da posizione che contiene gli attributi semplici latitudine e longitudine.

L’entità **MISURA\_COVID**, è costituita da quegli attributi che hanno valenza generale e che quindi appartengono a qualsiasi specializzazione: l’attributo totale\_casi, l’attributo composto note contenente un insieme di informazioni utili, costituito da note\_it e note\_en.

L’entità figlia **MISURA\_REGIONE** è costituita dagli attributi propriricoverati\_con\_sintomi, terapia\_intensiva, totale\_ospedalizzati, isolamento\_domiciliare, totale\_positivi, variazione\_totale\_positivi, nuovi\_positivi, dimessi\_guariti, decetudi, tamponi, casi\_testati, tutti atomici.

Per quanto riguarda l’entità **MISURA\_MOVIMENTI**, essaè costituita da un unico attributo strutturato variazioni contente l’insieme di attributi atomici: var\_svago\_vendita\_dettaglio, var\_farmacie\_alimentari, var\_parchi, var\_stazioni\_trasporto\_pubblico, var\_sedi\_lavoro, var\_zone\_residenziali.

Infine delle quattro **associazioni** Sito, Luogo\_Misurazione\_1, Luogo\_Misurazione\_2, Luogo\_Misurazione\_3,tutte tranne Sito, priva di attributi propri, posseggono l’attributo atomico *data*.

1. **Analisi dei dati**

L’analisi dei dati, si avvalora di Query, procedure e ad alcuni tool Grafici, per mostrare i dati e le relazioni tra essi.

* **Query e Grafici sulle variazioni di Spostamenti Regionali:**

**--Query parametrizzata per il report in formato CSV sui movimenti di una regione presa in INPUT**

**CREATE OR REPLACE PROCEDURE** report\_movimenti\_regione (regione\_in **IN**regioni.denominazione\_regione%TYPE)

**AS**

found\_regione **INTEGER**;

**CURSOR** cursore\_regione **IS** (

**SELECT**mov.codice\_regione, mov.data, mov.variazione\_svago\_vendita\_dettaglio, mov.variazione\_farmacie\_alimentari, mov.variazione\_parchi, mov.variazione\_stazioni\_trasporto\_pubblico, mov.variazione\_sedi\_lavoro, mov.variazione\_zone\_residenziali

**FROM**movimenti\_regioni mov **JOIN**regioni r **ON** mov.codice\_regione = r.codice\_regione

**WHERE LOWER**(denominazione\_regione) = **LOWER**(regione\_in)

);

riga\_movimenti movimenti\_regioni%**ROWTYPE**;

regione\_not\_found**EXCEPTION**;

**BEGIN**

**--Controllo che la regione sia presente**

**SELECT** count(distinct(codice\_regione)) **INTO**found\_regione

**FROM** regioni

**WHERE LOWER**(denominazione\_regione) = **LOWER**(regione\_in)**AND** codice\_regione**IN** (

**SELECT**distinct(codice\_regione)

**FROM** movimenti\_regioni

);

**IF** found\_regione <= 0 **THEN**

**RAISE** regione\_not\_found;

**END IF;**

**--posso stampare le regioni**

**OPEN** cursore\_regione;

**dbms\_output.put\_line**('DATI MOVIMENTI REGIONE '|| regione\_in);

**dbms\_output.put\_line**('codice\_regione,data,variazione\_svago\_vendita\_dettaglio,variazione\_farmacie\_alimentari,variazione\_parchi, variazione\_stazioni\_trasporto\_pubblico,variazione\_sedi\_lavoro,variazione\_zone\_residenziali');

**LOOP**

**FETCH** cursore\_regione **INTO** riga\_movimenti;

**EXIT WHEN** cursore\_regione**%NOTFOUND**;

**dbms\_output.put\_line**(riga\_movimenti.codice\_regione || ',' || riga\_movimenti.data || ',' || riga\_movimenti.variazione\_svago\_vendita\_dettaglio || ',' || riga\_movimenti.variazione\_farmacie\_alimentari|| ',' || riga\_movimenti.variazione\_parchi|| ',' || riga\_movimenti.variazione\_stazioni\_trasporto\_pubblico|| ',' || riga\_movimenti.variazione\_sedi\_lavoro|| ',' || riga\_movimenti.variazione\_zone\_residenziali );

**END LOOP;**

**CLOSE** cursore\_regione;

**EXCEPTION**

**WHEN** regione\_not\_found **THENdbms\_output.put\_line**('REGIONE NON PRESENTE');

**END** report\_movimenti\_regione;

* **MEDIA MOVIMENTI REGIONALI**

**------------------------------------------------------------------------------------------------------------------------------------------------------------------------------**

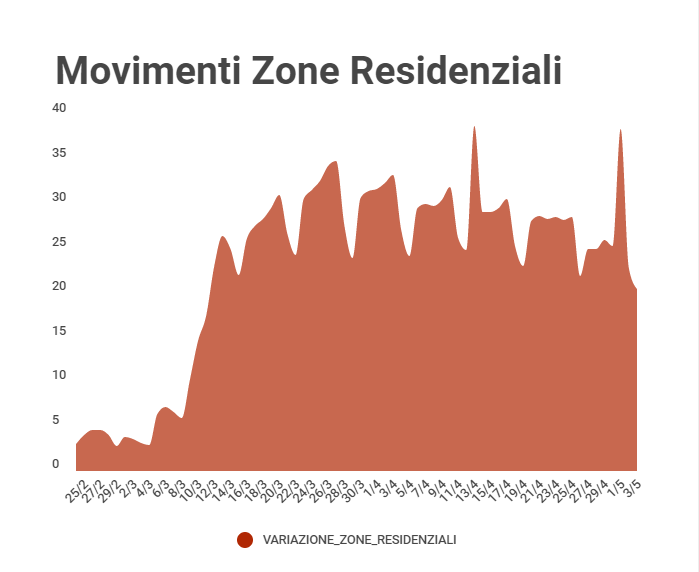
**--Media degli spostamenti degli italiani, verso una determinata categoria di luoghi**

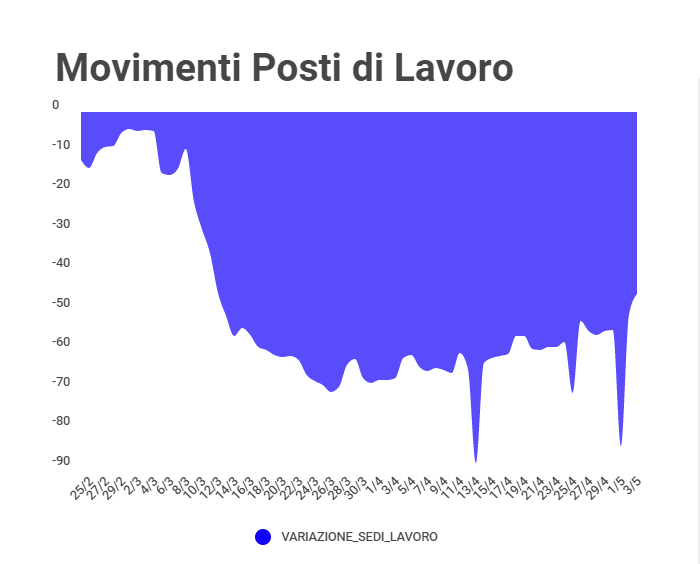
**SELECT** data**, AVG**(variazione\_\*\*\*\*)

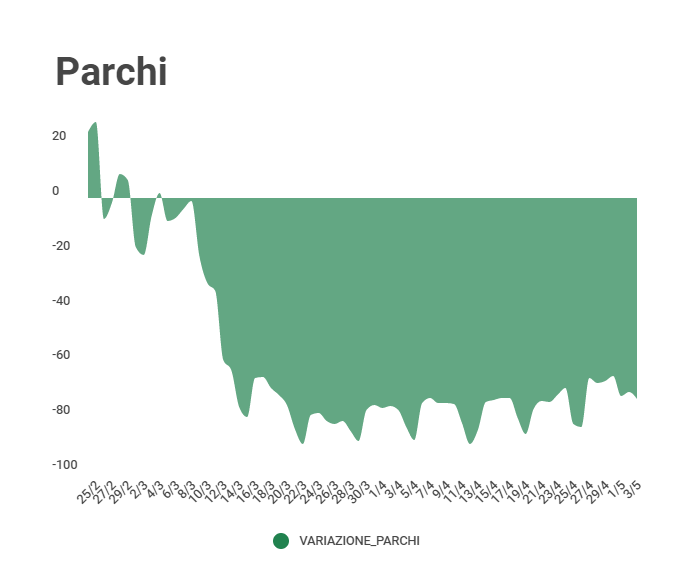
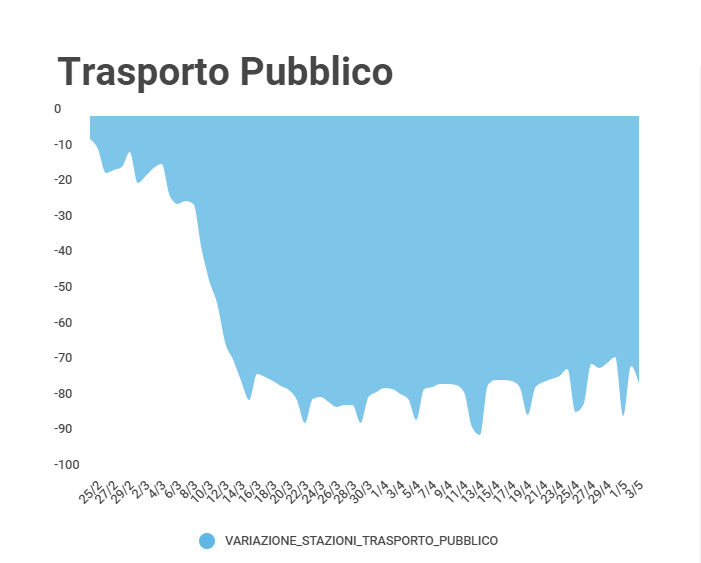
**FROM** movimenti\_regioni

**GROUP BY**data

**ORDER BY**data;

**------------------------------------------------------------------------------------------------------------------------------------------------------------------------------**





* **Query che ritorna il numero di contagi e la posizione per ogni provincia, alla data più recente**

**SELECT** p.longitudine, p.latitudine, mis.totale\_casi

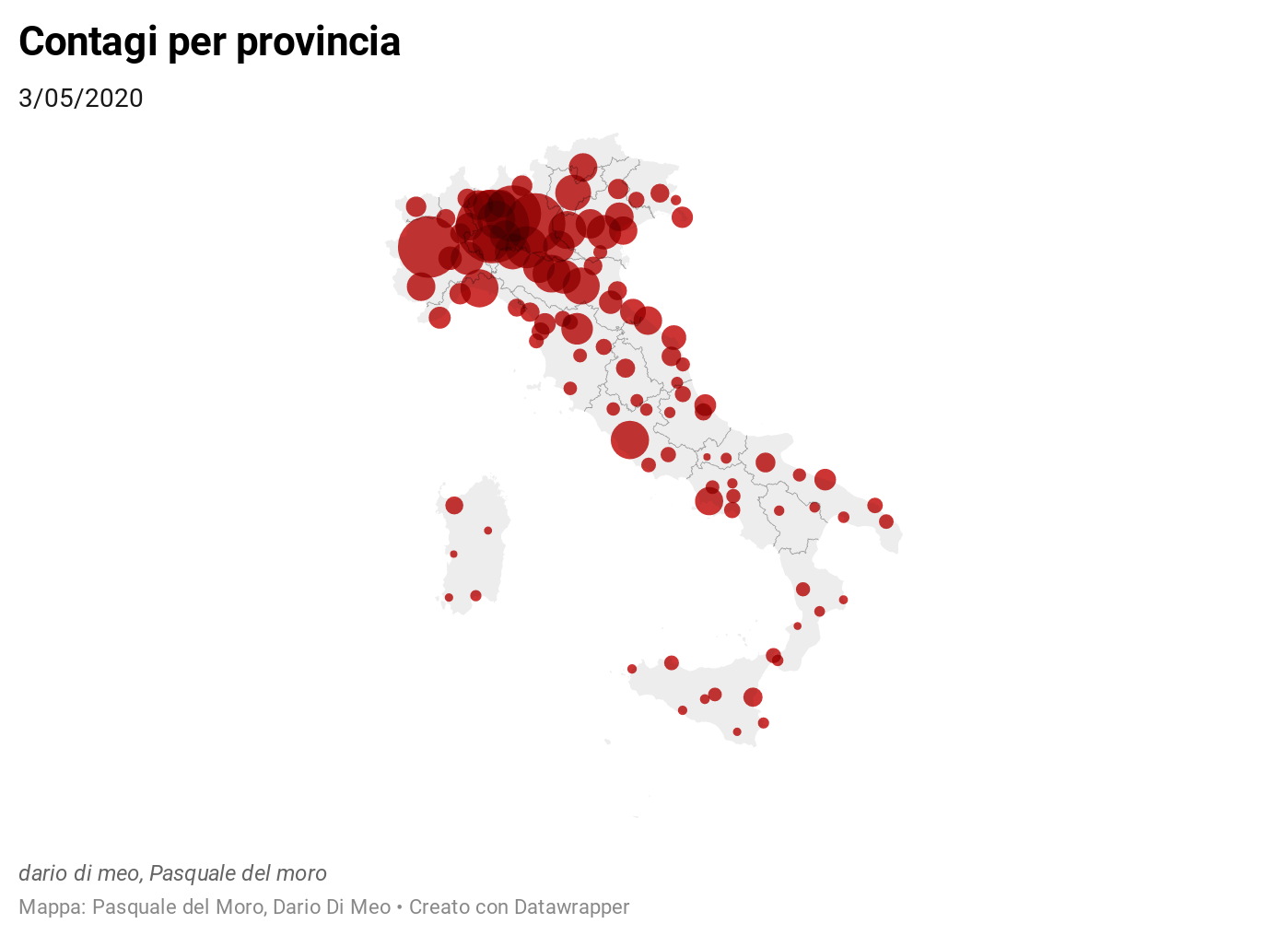
**FROM** misure\_province mis**JOIN** province p **ON**mis.codice\_provincia = p.codice\_provincia

**WHERE** data = (

**SELECT** max(data)

**FROM** misure\_province

);



* **Stored Procedure sul numero di tamponi e contagiati**

**--Genera uno script CSV sul rapporto tra positivi e tamponi, di tutte le regioni, data una data in INPUT, nel caso i tamponi**

**-- sono 0 restituisce NULL**

**CREATE OR REPLACE PROCEDURE** rapporto\_tamponi\_contagi\_regione ( data\_in **IN** misure\_regioni.data%**TYPE**)

**AS**

**TYPE** rec\_info **IS RECORD**(

data misure\_regioni.data%**TYPE**,

codice misure\_regioni.codice\_regione%**TYPE**,

positivi misure\_regioni.totale\_positivi%**TYPE**,

tamponi misure\_regioni.tamponi%**TYPE**

);

**CURSOR** cursor\_mis\_regioni **IS** (

**SELECT** data, codice\_regione, totale\_positivi, tamponi

**FROM** misure\_regioni

**WHERE** data = data\_in

);

output **rec\_info**;

data\_non\_presente **EXCEPTION**;

counter **INTEGER**;

**BEGIN**

**--controllo che la data sia presente**

**SELECT COUNT**(**DISTINCT** data) **INTO** counter

**FROM** misure\_regioni

**WHERE** data = data\_in;

**IF** counter = 0 **THEN**

**RAISE** data\_non\_presente;

**END IF**;

**--Posso continuare**

**dbms\_output.put\_line**('data;codice\_regione;rapporto\_tamponi\_contagi');

**OPEN** cursor\_mis\_regioni;

**LOOP**

**FETCH** cursor\_mis\_regioni **INTO** output;

**EXIT WHEN** cursor\_mis\_regioni%**NOTFOUND**;

**--Controllo che i tamponi non siano nulli**

**IF** output.tamponi = 0 **THEN**

dbms\_output.put\_line(output.data || ';' || output.codice|| ';NULL');

**ELSE**

dbms\_output.put\_line(output.data || ';' || output.codice|| ';' || output.positivi/output.tamponi);

**END IF;**

**END LOOP;**

**CLOSE** cursor\_mis\_regioni;

**EXCEPTION**

**WHEN** data\_non\_presente **THEN**

dbms\_output.put\_line('Data non presente');

**END** rapporto\_tamponi\_contagi\_regione;

* **Grafici animati su pygame**

Parte dei grafici saranno mostrati in seduta orale.

Riportiamo di seguito un'immagine dell'applicazione da noi sviluppata con la libreria grafica pygame mentre viene visualizzato un istogramma.

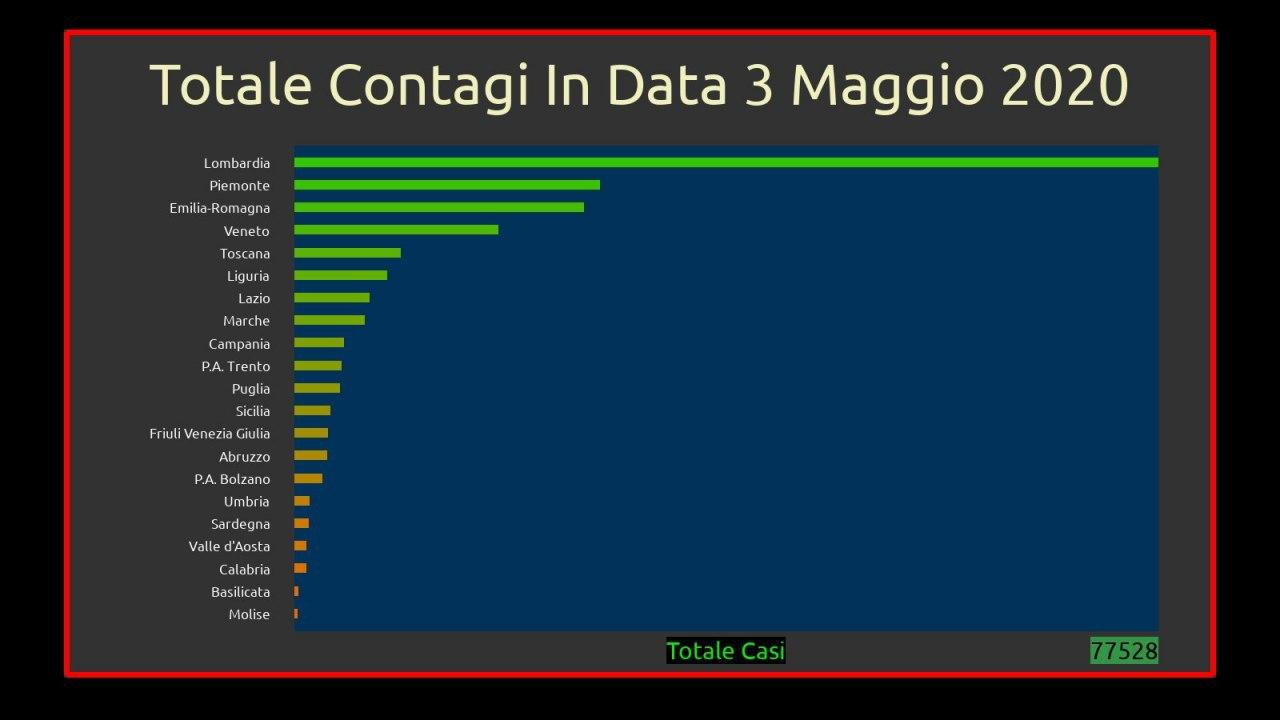


Foto esempio

1. Stored Procedure

Oltre alle Query parametrizzate usate nel paragrafo dedicato all’analisi dei dati, si è ritenuto opportuno creare alcune procedure per ricavare dati utili.

**Procedure e Funzioni**

-----------------------------------------------------------------------------------------------------------------

**--FUNCTION CHE RITORNA IL CODICE DELLA REGIONE COL MASSIMO DEI CONTAGI PER UNA CERTA DATA**

**CREATE OR REPLACE FUNCTION** get\_regione\_max\_contagi(data\_oss I**N** misure\_regioni.data%**TYPE**)

**RETURN** misure\_regioni.codice\_regione%**TYPE**

**AS**

cod\_max misure\_regioni.codice\_regione%**TYPE**;

**BEGIN**

**SELECT** codice\_regione **INTO** cod\_max

**FROM**misure\_regioni

**WHERE** data **=** data\_oss **AND** totale\_positivi **IN** (

**SELECTmax(**totale\_positivi**)**

**FROM**misure\_regioni

**WHERE** data **=** data\_oss

);

**RETURN** cod\_max;

**END** get\_regione\_max\_contagi;

**--FUNCTION CHE RITORNA IL CODICE DELLA PROVINCIA COL MASSIMO DEI CONTAGI DATO UNA DATA**

**CREATE OR REPLACE FUNCTION** get\_provincia\_max\_contagi(data\_oss **IN** misure\_province.data%**TYPE**)

**RETURN** misure\_province.codice\_provincia%**TYPE**

**AS**

cod\_max misure\_province.codice\_provincia%**TYPE**;

**BEGIN**

**SELECT** codice\_provincia **INTO** cod\_max

**FROM**misure\_province

**WHERE** data **=** data\_oss **AND** totale\_casi **IN** (

**SELECTmax(**totale\_casi**)**

**FROM**misure\_province

**WHERE** data **=** data\_oss

);

**RETURN** cod\_max;

**END** get\_provincia\_max\_contagi;

-----------------------------------------------------------------------------------------------------------------

**Trigger**

**--TRIGGER CHE RIEMPE AUTOMATICAMENTE LA TABELLA NORMALIZZATA DOPO GLI INSERT SULLA**

**--MASTER TABLE PROVINCE**

**CREATE OR REPLACE TRIGGER** trigger\_insert\_master\_province

**AFTER INSERT**

**ON** master\_table\_province

**FOR EACH ROW**

**BEGIN**

**INSERT INTO** misure\_province(data,codice\_provincia, totale\_casi, note\_it, note\_en) **VALUES** (**:NEW**.data,**:NEW**.codice\_provincia,**:NEW**.totale\_casi, **:NEW**note\_it, **:NEW**.note\_en);

**END;**

**--TRIGGER CHE RIEMPE AUTOMATICAMENTE LA TABELLA NORMALIZZATADOPO GLI INSERT SULLA**

**--MASTER TABLE REGIONI**

**CREATE OR REPLACE TRIGGER**trigger\_insert\_master\_regioni

**AFTER INSERT**

**ON** master\_table\_regioni

**FOR EACH ROW**

**BEGIN**

**INSERT INTO** misure\_regioni(data,

codice\_regione,

ricoverati\_con\_sintomi,

terapia\_intensiva,

totale\_ospedalizzati,

isolamento\_domiciliare,

totale\_positivi,

variazione\_totale\_positivi,

nuovi\_positivi,

dimessi\_guariti,

deceduti,

totale\_casi,

tamponi,

casi\_testati ,

note\_it ,

note\_en )

**VALUES** (**:NEW**.data,

**:NEW**.codice\_regione,

**:NEW**.ricoverati\_con\_sintomi,

**:NEW**.terapia\_intensiva,

**:NEW**.totale\_ospedalizzati,

**:NEW**.isolamento\_domiciliare,

**:NEW**.totale\_positivi,

**:NEW**.variazione\_totale\_positivi,

**:NEW**.nuovi\_positivi,

**:NEW**.dimessi\_guariti,

**:NEW**.deceduti,

**:NEW**.totale\_casi,

**:NEW**.tamponi,

**:NEW**.casi\_testati ,

**:NEW**.note\_it ,

**:NEW**.note\_en);

**END;**

1. **Ottimizzazione Base di dati**

L’ottimizzazione della base di dati, consiste nella creazione di **Indici** e **Viste**, per migliorare i tempi di esecuzione delle query.

Tramite le funzionalità di Cronologia SQL, su Oracle Live Sql**(**F8**)**,si può calcolare, in secondi, la durata di una query.

**Ad esempio:**

|  |
| --- |
| DURATA |
| 0.11s |

**SELECT** \*

**FROM** misure\_regioni 🡺

**WHERE** data ='3-mag-20';

**Indici**

Sono creati automaticamente sui campi chiave delle nostre tabelle. Generalmente le query sono basate su ricerche sui campi di tipo **DENOMINAZIONE**, cioè sui nomi delle regioni/province.

Per questo motivo si introducono due **INDICI SECONDARI** sui campi non chiave, denominazione.

---------------------------------------------------------------------------------------------------------

**--Indice su denominazione regione**

**CREATE INDEX** idx\_denominazione\_regione

**ON regioni**(denominazione\_regione);

**--Indice su denominazione Provincia**

**CREATE INDEX** idx\_denominazione\_provincia

**ON province(**denominazione\_provincia**);**

---------------------------------------------------------------------------------------------------------

|  |  |  |
| --- | --- | --- |
| Tabella | Prima dell’indice | Dopo l’indice |
| Regioni | 0.005s | 0.002s |
| Province | 0.005s | 0.002s |

Grazie a questi due indici si nota un notevole decremento del tempo di esecuzione delle query, quantificato dallo strumento di analisi di Oracle Live Sql.

Valori ottenuti dalle seguenti Query di **Testing**:

**SELECT** \* **SELECT** \*

**FROM** regioni **FROM** province

**WHERE** denominazione\_regione = 'Campania'; **WHERE** denominazione\_provincia = 'Napoli';

**Note:**

Il calcolo del tempo di esecuzione è **Machine dipendant**, quindi è solo una stima del tempo necessario. Inoltre i tempi calcolati sono frutto di una **media aritmetica** su più prove ripetute.

**Viste**

Le viste (non materializzate), sono Query memorizzate che quando vengono chiamate sono automaticamente eseguite per costruire una tabella temporanea su cui effettuare operazioni SQL.

Le viste vengono principalmente create per:

* Ridurre la complessità delle istruzioni sql (snellire il codice)
* Mostrare solo specifici attributi all’ utente utilizzatore della vista

Per questi motivi, si cercano**Query** frequentemente utilizzate sia per l’analisi dei dati, sia per gli eventuali utenti che utilizzano la base dati come **supporto** per una loro applicazione.

Si è pensato di creare una vista per la visualizzazione, alla **data più recente**, dei contagi per ogni **provincia**, alla quale è stata aggiunta la colonna *denominazione\_provincia* per facilitare eventuali ricerche sul campo.

**--Vista dati più recenti sui contagi per provincia**

**CREATE VIEW** last\_contagi\_province **AS**(

**SELECT** mis.data, mis.codice\_provincia, p.denominazione\_provincia, mis.totale\_casi

**FROM** misure\_province mis **JOIN**province p **ON** mis.codice\_provincia = p.codice\_provincia

**WHERE** mis.data **IN**(

**SELECT** max(data)

**FROM** misure\_province

)

);

**Viste Materializzate**

Le viste materializzate, a differenza delle viste classiche, sono uno **Snapshot** di una particolare Query.Quindi utilizzate per dati statici che mutano **molto raramente**, inoltre, sono molto più **efficienti** delle viste normali, in quanto non devono essere ricreate ad ogni utilizzo.

Una vista utile, è una sui dati riguardanti i posti letto presenti per ogni regione e il numero di positivi di ogni giorno.

**--Vista materializzata sui dati regionali mettendo in relazione i posti letto e i contagi**

**CREATE MATERIALIZED VIEW** posti\_regione\_positivi **AS**(

**SELECT**mis.data, r.codice\_regione, r.posti\_letto, r.posti\_terapia\_intensiva, mis.totale\_positivi

**FROM**misure\_regioni mis **JOIN**regioni r **ON** mis.codice\_regione = r.codice\_regione

);

**Appendice**

1. **Creazione Master Table**

Le Master Table sono tabelle non normalizzate il cui unico scopo è di essere un supporto iniziale, per la memorizzazione diretta dei dati forniti dalla Protezione Civile Italiana sul Covid. Dunque essecontengono i dati grezzi, su cui grazie alla normalizzazione e all’aggiunta di attributi si ottiene la Base Di Dati finale.

**CODICE SQL CREATE MASTER TABLE:**

**CREATE TABLE** master\_table\_province

(

data **DATE**,

stato **CHAR(3),**

codice\_regione **NUMBER(2),**

denominazione\_regione **VARCHAR2(200),**

codice\_provincia **NUMBER(3) NOT NULL**,

denominazione\_provincia **VARCHAR2(200),**

sigla\_provincia **CHAR(2),**

latitudine **NUMBER,**

longitudine **NUMBER,**

totale\_casi **INTEGER,**

note\_it **CLOB,**

note\_en **CLOB**

);

**ALTER TABLE** master\_table\_province **ADD CONSTRAINT** PK\_MASTER\_PROVINCE **PRIMARY KEY** (data, codice\_provincia);

**CREATE TABLE** master\_table\_regioni

( data **DATE**,

stato **CHAR(3),**

codice\_regione **NUMBER(2),**

denominazione\_regione **VARCHAR2(200),**

latitudine **NUMBER,**

longitudine **NUMBER,**

ricoverati\_con\_sintomi **INTEGER,**

terapia\_intensiva **INTEGER,**

totale\_ospedalizzati **INTEGER,**

isolamento\_domiciliare **INTEGER,**

totale\_positivi **INTEGER,**

variazione\_totale\_positivi **INTEGER,**

nuovi\_positivi **INTEGER,**

dimessi\_guariti **INTEGER**,

deceduti **INTEGER,**

totale\_casi **INTEGER,**

tamponi **INTEGER,**

casi\_testati **INTEGER,**

note\_it **CLOB,**

note\_en **CLOB**

);

**ALTER TABLE** master\_table\_regioni **ADD CONSTRAINT** PK\_MASTER\_REGIONI **PRIMARY KEY** (data, codice\_regione);

1. **Fonti e siti web utili**

**Dati:**

Dati Protezione Civile: <https://github.com/pcm-dpc/COVID-19>

Posti letto e posti terapia intensiva: <http://www.healthdata.org/covid/data-downloads>

Numero di abitanti, Densità abitativa: <http://dati.istat.it/>

Variazione spostamenti regionali: <https://www.google.com/covid19/mobility/>

**Siti web utili per grafici:**

<https://app.datawrapper.de/>

<https://infogram.com/>