



Progetto di
Basi di Dati

Professore
Elio Masciari

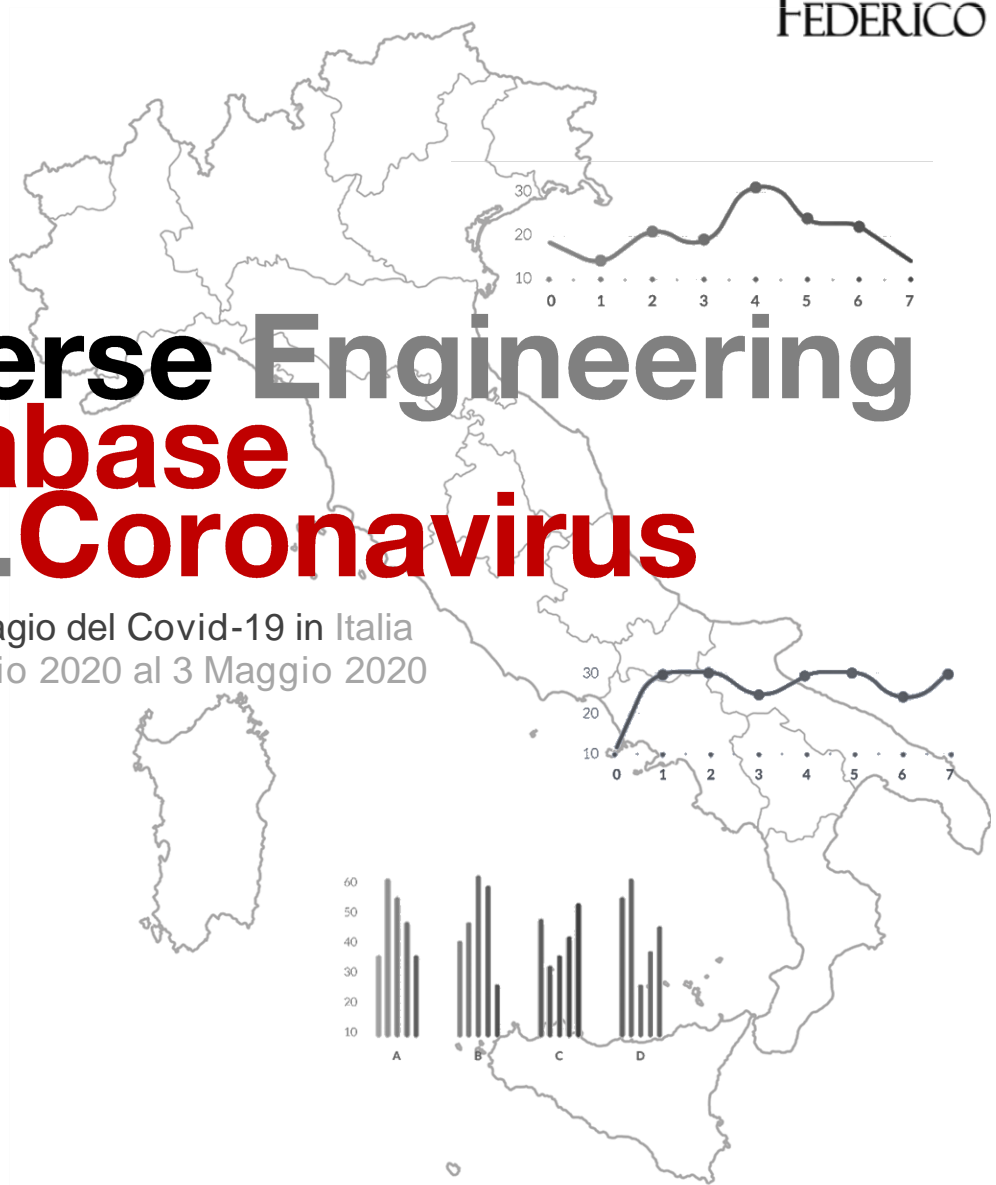
I A.A 2019/2020



UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II

Reverse Engineering Database Dati. Coronavirus

I dati del contagio del Covid-19 in Italia
Dal 25 febbraio 2020 al 3 Maggio 2020



Autori

Antonio Romano N46004321
Giuseppe Riccio N46004297

Corso di Laurea di
Ingegneria Informatica

L'indice

La traccia	4
Prefazione.....	5
Cos' è il covid-19?	5
Capitolo 1	6
La tabella master	6
Capitolo 2 Raccolta e Analisi dei requisiti dalla tabella Master	9
2.1 Specifiche sulle operazioni.....	11
2.2 Vincoli tecnologici e criteri di sicurezza dei dati	12
Capitolo 3 La progettazione concettuale.....	13
3.1 Lo schema ER portante ed ER con cardinalità.....	13
3.2 Glossario dei termini che verranno utilizzati nel database.....	15
3.3 Possibile raffinamento dello schema ER finale e le possibili trasformazioni o traduzioni	18
3.4 ER finale della base di dati	19
3.5 Schema logico della base di dati.....	21
3.6 Qualità dello schema logico e normalizzazione	22
3.6.1 - Comandi ddl	24
Capitolo 4 La progettazione fisica	27
4.1 Condizioni operative del database	27
4.2 Dimensionamento fisico della base di dati.....	27
4.3 Configurazione del DB server e dimensionamento totale della base di dati.....	30
Capitolo 5 Creazione della base di dati	31
5.1 Query and Reporting	34
5.2 Guida alla lettura dei dati attraverso query	46
Capitolo 6 Livello dati	50
6.2 Guida alla lettura dei dati attraverso procedure PL/SQL	50
6.3 Il calcolo dell'R0 dell'epidemia	58
6.4 Il calcolo del tasso di letalità dell'epidemia	61
6.5 Automatizzazione calcolo delle variazioni percentuali rispetto al giorno precedente attraverso l'uso dei trigger	64
6.6 Trigger per la sicurezza dei dati	65
Capitolo 7 Il livello Applicazione.....	67
Capitolo 8 Il livello di presentazione.....	68
8.1 Indice sulle relazioni regioni, province, stati	68
8.2 Le view sul DB_COVID19	69

8.3 Utilizzo delle view	70
8.4 Una possibile interfaccia web.....	73
Appendice Simulatore della base di dati DB_COVID19.....	74
Sitografia e Bibliografia.....	76

“

Nulla si ottiene senza sacrificio e senza coraggio.
Chi ha ragione ed è capace di soffrire alla fine vince.

M.Gandhi

Lavoro svolto con dedizione e pazienza
ai tempi del **CoronaVirus.**

Gli Autori

La traccia

Progetto di Basi di Dati
a.a. 2019/2020
Prof. Elio Masciari

Dato il **DATASET**, costituito dall'insieme dei file in formato CSV scaricabili da <https://github.com/pcm-dpc/COVID-19/tree/master/dati-province>, e relativo all'andamento del contagio del virus COVID-19 nelle province Italiane, si effettui attraverso l'ambiente ORACLE LIVE SQL (o mediante la distribuzione Oracle XE 18c):

1) La creazione di una **tabella master** (con lo stesso schema dei file in formato CSV) ed il relativo popolamento con i dati del contagio per tutte le province italiane dal 25/02/2020 al 03/05/2020 (utilizzare una qualsiasi strumento di conversione da CSV a SQL come ad esempio il tool <https://www.convertcsv.com/csv-to-sql.htm>);

2) La verifica della **3NF** per lo schema della tabella precedentemente istanziata e l'eventuale decomposizione con la definizione di tutti i vincoli (mediante comandi DDL);

3) L'arricchimento dello schema (medianti comandi di **ALTER TABLE**) ottenuto con ulteriori informazioni utili all'analisi del fenomeno, ad esempio:

a. per ogni data il numero di morti, il numero di ricoveri in strutture ospedaliere, il numero di ricoveri in terapia intensiva, etc.

b. per ogni regione il numero di abitanti, la densità abitativa, superficie in km2, numero di autostrade e strade statali, numero di aeroporti e stazioni, etc.;

c. per ogni provincia il numero il numero di abitanti, la densità abitativa, il numero di scuole, alberghi/strutture recettive, il numero di ospedali, il numero di spostamenti intra e extra provincia, etc. d. etc.

4) L'individuazione, attraverso un processo di **REVERSE ENGINEERING**, di un possibile schema concettuale E/R della base di dati;

5) La specifica in SQL di una serie di query utili all'analisi dell'andamento del contagio del COVID-19 (e.g., *numero di contagi per provincia in una determinata finestra temporale, regione col il maggior numero di contagi per densità abitativa, etc.*), con un'eventuale visualizzazione grafica dei risultati (es. attraverso l'utilizzo di MS Excel);

6) La specifica in PL/SQL di una serie di procedura/trigger che possano consentire un'analisi più flessibile ed eventuali aggiornamenti automatici della base di dati qualora vogliano essere importati dati successivi al 03/5/2020, ad esempio:

a. STORED PROCEDURE aventi come paramenti di ingresso le differenti variabili dell'analisi (es. giorno o finestra temporale di analisi, provincia o regione da analizzare, etc.);

b. TRIGGER che all'atto di inserimenti nella tabella master esegue in automatico il popolamento delle tabelle dello schema normalizzato;

7) La definizione di viste sui dati ed indici che possono essere utili per migliorare i tempi di esecuzione delle query.

Prefazione

La seguente trattazione è mirata al Reverse Engineering di una complessa base di dati sull'andamento del contagio del Coronavirus COVID-19 a partire dai primi contagi in Italia: 25 febbraio 2020 fino al 3 Maggio 2020.

La decostruzione avverrà sulla base di una **tabella master** comprendendo, riscrivendo e ricostruendo l'architettura del database con un ulteriore studio di analisi e sviluppo del database con un'analisi più approfondita eliminando anche ulteriori ridondanze attraverso il processo di normalizzazione.

Lo sviluppo della base di dati sarà poi espanso nel raccogliere tutti dati necessari per determinare l'andamento del contagio nelle province, regioni e nel mondo.

Il nostro obiettivo sarà dunque la ricerca di tutte le possibili implementazioni e ottimizzazioni, per rendere questo database non fine a sé stesso, o per questo specifico contesto, ma sfruttabile anche per futuri usi, di diverso genere. Fornendo un opportuno strumento di gestione del territorio e supporto alle decisioni da prendere per gestire una situazione di profonda emergenza come questa del Coronavirus, ma anche un'emergenza, ad esempio, di carattere economico-sociale.

Tuttavia, si è pensato anche di documentare e di coinvolgere l'uso e la creazione della base di dati supportando il lettore con una **guida alla lettura dei dati** usando un linguaggio semplice e basilare e non noioso. Infatti, si è cercato di seguire la **traccia** assegnata per la realizzazione del progetto usando e aggiungendo **nostre idee** affinché diventasse un progetto completo arricchendolo inoltre di **link ipertestuali** collegati al codice di realizzazione delle base di dati realizzata su **DataGrip 20.1** connesso ad un **DBMS ORACLE 18c Express Edition**.

DEFINIZIONE DEL REVERSE ENGINEERING

“Reverse engineering: consiste nella ricostruzione di un prodotto già esistente, in questo caso di un software. Il prodotto viene “smontato” per comprenderne l'architettura, la struttura e il funzionamento. Lo scopo del reverse engineering in ambito software è quello di riprodurre il codice di un programma esistente. In questo modo è possibile ottimizzare il software correggendone gli errori di funzionamento, analizzare i programmi della concorrenza o sviluppare nuovi prodotti.”

Cos' è il covid-19?

Si chiama 2019-nCoV e come tutti i coronavirus è un virus a RNA provvisto di envelope. Questa nuova minaccia, che ha allarmato la Cina durante gli ultimi giorni del 2019, è molto simile a due vecchie conoscenze: la Mers, la Sindrome respiratoria mediorientale, e la Sars, ovvero la Sindrome respiratoria acuta grave. L'epicentro della sua diffusione è Wuhan, luogo dove hanno cominciato a diffondersi i primi casi: al 21 gennaio del 2020, secondo i dati diffusi dalla Cina, se ne sono registrati ben 223, di cui 218 in Cina, 2 in Thailandia, 1 in Giappone e l'ultimo in Corea del Sud per poi giungere a numeri impressionanti. Ormai, nel tardo pomeriggio di ogni giorno, siamo abituati a vedere il capo della Protezione civile, e commissario straordinario per l'emergenza Covid-19 a rilasciare in conferenza stampa una serie di numeri relativi alla diffusione del nuovo coronavirus (Sars-CoV-2) nel nostro Paese. I dati su casi positivi, guariti e deceduti sono consultabili nella realizzazione della base di dati in esame con le statistiche regionali sul numero dei test effettuati e sul numero dei ricoverati con sintomi, di chi è in terapia intensiva o in isolamento domiciliare e da questi dati ufficiali, forniti dal ministero della Salute, si ritiene di poter monitorare, in modo preciso, se l'epidemia si stia espandendo o meno, dove, e in che modo. (...).

Capitolo 1

La tabella master

Il dataset fornito da <https://github.com/pcm-dpc/COVID-19> è una rappresentazione della base di dati direzionale, indicato anche col termine **Data WareHouse (DWH)**.

La prima caratteristica del database è che esso è una collezione di dati, o meglio un deposito di dati orientati al soggetto, ovvero orientato verso gli utenti curiosi, enti di ricerca, televisione ecc.

Un'altra caratteristica rilevante è la tempo-varianza, tutti i dati contenuti nel DWH si riferiscono ad un preciso arco temporale, una volta che sono stati correttamente inseriti non possono essere modificati, quindi, i dati possono essere solo **caricati ed acceduti**. Si può dire che esso è uno strumento sofisticato per arrivare ad effettuare analisi e prendere opportune decisioni.

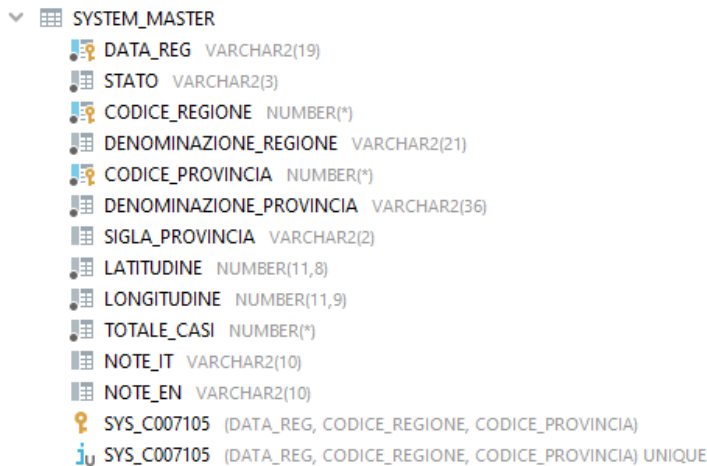
Il Dataset → Tutti i dati del contagio in Italia dal 25 Febbraio 2020 al 3 Maggio 2020

..omissis...

DATA_REG	STATO	CODICE...	DENOMINAZIONE...	CODICE_PROV...	DENOMINAZIONE_PRO...	SIGLA...	LATI...	LONG...	TOTALE_CASI	NOTE_IT	NOTE_EN
1	2020-03-08T18:00:00	ITA	13	Abruzzo	69	Chieti	CH	42.35183167	14.167545740	4	<null>
2	2020-03-08T18:00:00	ITA	13	Abruzzo	66	L'Aquila	AQ	42.35122196	13.398438230	1	<null>
3	2020-03-08T18:00:00	ITA	13	Abruzzo	68	Pescara	PE	42.46458398	14.213648220	8	<null>
4	2020-03-08T18:00:00	ITA	13	Abruzzo	67	Teramo	TE	42.65891778	13.784399710	4	<null>
5	2020-03-08T18:00:00	ITA	13	Abruzzo	979	In fase di definizione/a...	<null>	0.00000000	0.00000000	0.00000000	<null>
6	2020-03-08T18:00:00	ITA	17	Basilicata	77	Matera	MT	40.66751177	16.597924420	2	<null>
7	2020-03-08T18:00:00	ITA	17	Basilicata	76	Potenza	PZ	40.63947052	15.885148340	2	<null>
8	2020-03-08T18:00:00	ITA	17	Basilicata	980	In fase di definizione/a...	<null>	0.00000000	0.00000000	0.00000000	<null>
9	2020-03-08T18:00:00	ITA	18	Calabria	79	Catanzaro	CZ	38.98597598	16.594401940	2	<null>
10	2020-03-08T18:00:00	ITA	18	Calabria	78	Cosenza	CS	39.29388681	16.256896920	1	<null>
11	2020-03-08T18:00:00	ITA	18	Calabria	101	Crotone	KR	39.08836878	17.125388640	0.00000000	<null>
12	2020-03-08T18:00:00	ITA	18	Calabria	80	Reggio di Calabria	RC	38.10922769	15.643452700	1	<null>
13	2020-03-08T18:00:00	ITA	18	Calabria	102	Vibo Valentia	VV	38.67624147	16.181574140	0.00000000	<null>
14	2020-03-08T18:00:00	ITA	18	Calabria	982	In fase di definizione/a...	<null>	0.00000000	0.00000000	5	<null>
15	2020-03-08T18:00:00	ITA	15	Campania	64	Avellino	AV	40.91484699	14.795288030	3	<null>
16	2020-03-08T18:00:00	ITA	15	Campania	62	Benevento	BN	41.12969987	14.781516830	4	<null>
17	2020-03-08T18:00:00	ITA	15	Campania	61	Caserta	CE	41.07465878	14.332406460	28	<null>
18	2020-03-08T18:00:00	ITA	15	Campania	63	Napoli	NA	40.83956555	14.250849840	45	<null>
19	2020-03-08T18:00:00	ITA	15	Campania	65	Salerno	SA	40.67821961	14.759402600	15	<null>
20	2020-03-08T18:00:00	ITA	15	Campania	983	In fase di definizione/a...	<null>	0.00000000	0.00000000	6	<null>
21	2020-03-08T18:00:00	ITA	8	Emilia-Romagna	37	Bologna	BO	44.49436681	11.341728800	62	<null>
22	2020-03-08T18:00:00	ITA	8	Emilia-Romagna	38	Ferrara	FE	44.83599805	11.618689340	6	<null>
23	2020-03-08T18:00:00	ITA	8	Emilia-Romagna	40	Forlì-Cesena	FC	44.22268559	12.040686080	15	<null>
24	2020-03-08T18:00:00	ITA	8	Emilia-Romagna	36	Modena	MO	44.64600009	10.926154870	97	<null>
25	2020-03-08T18:00:00	ITA	8	Emilia-Romagna	34	Parma	PR	44.80187394	10.328349850	276	<null>
26	2020-03-08T18:00:00	ITA	8	Emilia-Romagna	33	Piacenza	PC	45.05193462	9.692632596	528	<null>
27	2020-03-08T18:00:00	ITA	8	Emilia-Romagna	39	Ravenna	RA	44.41722493	12.199139360	13	<null>
28	2020-03-08T18:00:00	ITA	8	Emilia-Romagna	35	Reggio nell'Emilia	RE	44.69735289	10.638079730	70	<null>
29	2020-03-08T18:00:00	ITA	8	Emilia-Romagna	99	Rimini	RN	44.06890807	12.565629500	113	<null>
30	2020-03-08T18:00:00	ITA	8	Emilia-Romagna	984	In fase di definizione/a...	<null>	0.00000000	0.00000000	0.00000000	<null>
31	2020-03-08T18:00:00	ITA	6	Friuli Venezia Giulia	31	Gorizia	GO	45.94149817	13.622125020	6	<null>
32	2020-03-08T18:00:00	ITA	6	Friuli Venezia Giulia	93	Pordenone	PN	45.95443546	12.668029090	2	<null>
33	2020-03-08T18:00:00	ITA	6	Friuli Venezia Giulia	32	Trieste	TS	45.64943548	13.768136490	25	<null>
34	2020-03-08T18:00:00	ITA	6	Friuli Venezia Giulia	30	Udine	UD	46.06255516	13.234838300	24	<null>
35	2020-03-08T18:00:00	ITA	6	Friuli Venezia Giulia	985	In fase di definizione/a...	<null>	0.00000000	0.00000000	0.00000000	<null>
36	2020-03-08T18:00:00	ITA	12	Lazio	60	Frosinone	FR	41.63946569	13.351171610	2	<null>
37	2020-03-08T18:00:00	ITA	12	Lazio	59	Latina	LT	41.46759465	12.983684820	6	<null>
38	2020-03-08T18:00:00	ITA	12	Lazio	57	Rieti	RI	42.40848444	12.862059390	0.00000000	<null>
39	2020-03-08T18:00:00	ITA	12	Lazio	58	Roma	RM	41.89277844	12.483646720	77	<null>
40	2020-03-08T18:00:00	ITA	12	Lazio	56	Viterbo	VT	42.41738280	12.184734160	2	<null>
41	2020-03-08T18:00:00	ITA	12	Lazio	986	In fase di definizione/a...	<null>	0.00000000	0.00000000	0.00000000	<null>
42	2020-03-08T18:00:00	ITA	7	Liguria	10	Genova	GE	44.41149314	8.932699200	25	<null>
43	2020-03-08T18:00:00	ITA	7	Liguria	8	Imperia	IM	43.88570648	8.827850298	10	<null>
44	2020-03-08T18:00:00	ITA	7	Liguria	11	La Spezia	SP	44.10704991	9.828189700	11	<null>
45	2020-03-08T18:00:00	ITA	7	Liguria	9	Savona	SV	44.30750461	8.481188654	25	<null>
46	2020-03-08T18:00:00	ITA	7	Liguria	987	In fase di definizione/a...	<null>	0.00000000	0.00000000	7	<null>
47	2020-03-08T18:00:00	ITA	3	Lombardia	16	Bergamo	BG	45.69441368	9.668424528	997	<null>
48	2020-03-08T18:00:00	ITA	3	Lombardia	17	Brescia	BS	45.53993852	10.219183230	501	<null>
49	2020-03-08T18:00:00	ITA	3	Lombardia	13	Como	CO	45.80999128	9.885159546	27	<null>
50	2020-03-08T18:00:00	ITA	3	Lombardia	19	Cremona	CR	45.13336675	10.824208650	665	<null>
51	2020-03-08T18:00:00	ITA	3	Lombardia	97	Lecco	LC	45.85575781	9.393392246	53	<null>
52	2020-03-08T18:00:00	ITA	3	Lombardia	98	Lodi	LO	45.31448693	9.503728769	853	<null>
53	2020-03-08T18:00:00	ITA	3	Lombardia	20	Mantova	MN	45.15726772	10.792773630	56	<null>
54	2020-03-08T18:00:00	ITA	3	Lombardia	15	Milano	MI	45.46679489	9.190347404	406	<null>
55	2020-03-08T18:00:00	ITA	3	Lombardia	108	Monza e della Brianza	MB	45.58439843	9.273582472	59	<null>
56	2020-03-08T18:00:00	ITA	3	Lombardia	18	Pavia	PV	45.18508264	9.160157191	243	<null>
57	2020-03-08T18:00:00	ITA	3	Lombardia	14	Sondrio	SO	46.17899261	9.871474890	6	<null>
58	2020-03-08T18:00:00	ITA	3	Lombardia	12	Varese	VA	45.81781477	8.822868344	32	<null>
59	2020-03-08T18:00:00	ITA	3	Lombardia	988	In fase di definizione/a...	<null>	0.00000000	0.00000000	291	<null>
60	2020-03-08T18:00:00	ITA	11	Marche	42	Ancona	AN	43.61759753	13.518875300	54	<null>
61	2020-03-08T18:00:00	ITA	11	Marche	44	Ascoli Piceno	AP	42.85322384	13.576911270	0.00000000	<null>
62	2020-03-08T18:00:00	ITA	11	Marche	109	Fermo	FM	43.16058534	13.718395350	5	<null>
63	2020-03-08T18:00:00	ITA	11	Marche	43	Macerata	MC	43.38023926	13.453071820	9	<null>
64	2020-03-08T18:00:00	ITA	11	Marche	41	Pesaro e Urbino	PU	43.91014821	12.913459890	204	<null>
65	2020-03-08T18:00:00	ITA	11	Marche	989	In fase di definizione/a...	<null>	0.00000000	0.00000000	0.00000000	<null>
66	2020-03-08T18:00:00	ITA	14	Molise	70	Campobasso	CB	41.55774754	14.659160510	14	<null>
67	2020-03-08T18:00:00	ITA	14	Molise	94	Isernia	IS	41.58800826	14.225754070	0.00000000	<null>
68	2020-03-08T18:00:00	ITA	14	Molise	990	In fase di definizione/a...	<null>	0.00000000	0.00000000	0.00000000	<null>
69	2020-03-08T18:00:00	ITA	4	P.A. Bolzano	21	Bolzano	BZ	46.49933453	11.356624220	9	<null>
70	2020-03-08T18:00:00	ITA	4	P.A. Bolzano	981	In fase di definizione/a...	<null>	0.00000000	0.00000000	0.00000000	<null>
71	2020-03-08T18:00:00	ITA	4	P.A. Bolzano	22	Trento	TN	46.06893511	11.121238970	23	<null>
72	2020-03-08T18:00:00	ITA	4	P.A. Trento	996	In fase di definizione/a...	<null>	0.00000000	0.00000000	0.00000000	<null>
73	2020-03-08T18:00:00	ITA	1	Piemonte	6	Alessandria	AL	44.91297351	8.615401155	60	<null>
74	2020-03-08T18:00:00	ITA	1	Piemonte	5	Asti	AT	44.89912921	8.284142547	58	<null>
75	2020-03-08T18:00:00	ITA	1	Piemonte	96	Biella	BI	45.56651120	8.854082167	19	<null>

..omissis...

La tabella master fornisce il **numero dei casi** presenti sul territorio italiano diviso in **province**. La composizione(schema) della tabella:



SYSTEM_MASTER	
DATA_REG	VARCHAR2(19)
STATO	VARCHAR2(3)
CODICE_REGIONE	NUMBER(*)
DENOMINAZIONE_REGIONE	VARCHAR2(21)
CODICE_PROVINCIA	NUMBER(*)
DENOMINAZIONE_PROVINCIA	VARCHAR2(36)
SIGLA_PROVINCIA	VARCHAR2(2)
LATITUDINE	NUMBER(11,8)
LONGITUDINE	NUMBER(11,9)
TOTALE_CASI	NUMBER(*)
NOTE_IT	VARCHAR2(10)
NOTE_EN	VARCHAR2(10)
SYS_C007105	(DATA_REG, CODICE_REGIONE, CODICE_PROVINCIA)
SYS_C007105	(DATA_REG, CODICE_REGIONE, CODICE_PROVINCIA) UNIQUE

FIGURA 1 | Schema

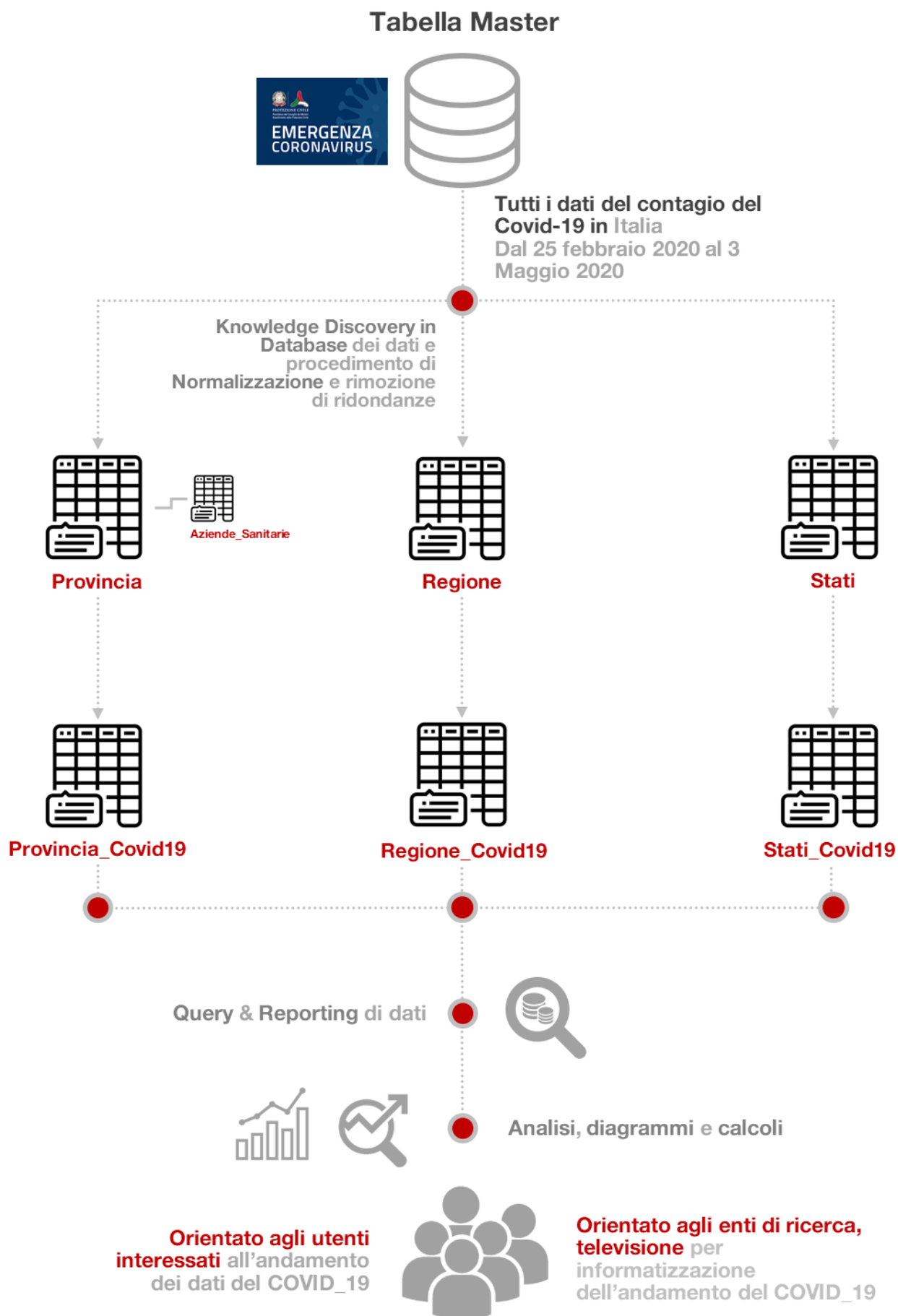
È possibile visualizzare e provare le seguenti definizioni in SQL [Cliccando qui](#)
Oppure al seguente link <https://paste.ee/r/PvZ7n/0>

In questa prima fase, il principale obiettivo, è stato quello dell'organizzazione dei file presenti nella cartella GitHub, data la lista di file.csv relativi ai rilievi per provincia dei giorni dal 25 Febbraio al 3 Maggio, è stato necessario il **merge** di tali file in un unico .csv, in questo modo la conversione sul sito <https://www.convertcsv.com/csv-to-sql.htm>, è stata resa più facile e veloce. Dopo fatto ciò, abbiamo creato i comandi DDL per definire la tabella **master** su DataGrip, il risultato finale è stato quello mostrato sopra.

Durante il procedimento di integrazione dei dati si è dovuto discutere della problematica ad una possibile anomalia sulla **DATA** di registrazione che, da come si evince dalla **Figura 1** sopra è del tipo varchar, la scelta di questo tipo è stata presa nell'ottica di un'ottimizzazione delle operazioni da fare sulla tabella. Infatti, se fosse stato scelto il tipo date, innanzitutto sarebbe stato necessario la modifica di tutte le date presenti nei file originali, ma cosa ancor più importante non sarebbero stati possibili confronti con gli usuali operatori (<, >, =), infatti, per il tipo date è possibile effettuare confronti unicamente convertendolo prima in **char** tramite il comando **to_char**.

Dalla **Figura 2, pag.8** si può notare il procedimento di reverse engineering, ovvero, partendo dalla tabella master e con un doveroso studio di *Knowledge Discovery in Database (KDD)* dei dati e procedendo con la **normalizzazione** dei dati (**Cap.3 p. 3.6**) si è decomposta la base di dati in **Provincia_COVID19**, **Regione_COVID19**, **Stati_COVID19** e le relative generalizzazioni (**Cap.3**). Dopo quindi la definizione delle informazioni si è passata alla stesura delle query (**Cap.5**) e reporting (**Cap.6**). Tutti i risultati dunque sono rivolti ad una precisa community, enti di ricerca, enti governativi, enti informativi utili ad informare o a prendere decisioni governative sul territorio quindi attuando un procedimento di *Decisions Support System (DSS)*.

FIGURA 2 | Schema riassuntivo del progetto



Capitolo 2

Raccolta e Analisi dei requisiti dalla tabella Master

La raccolta e l'analisi dei requisiti rappresenta la fase in cui vengono definiti i desideri e le necessità a cui il database deve rispondere al fine di rispondere alle domande degli utenti in maniera semplice ed efficiente. Questa fase è tra le più importanti perché studia le proprietà che il database deve avere e fornisce ai progettisti una specifica, chiara ed intuitiva, da tradurre nel relativo progetto concettuale, logico e successivamente fisico del database.

Il **database** fornisce a chiunque sia interessato la possibilità di informare enti di ricerca, enti governativi, enti informativi mettendo a disposizione i dati raccolti, utili ai soli fini comunicativi e di informazione. Il database tiene traccia di tutti i dati esposti dalla **Protezione Civile** e dunque necessario conservare le usuali informazioni delle **regioni**:

- *la data*, indica il giorno interessato;
- *stato*, indica lo stato interessato;
- *la regione*, indica la regione interessata con la relativa chiave di identificazione;
- *latitudine e longitudine* determina la posizione delle zone interessate;
- *ricoverati con sintomi*, determina le persone ricoverati con i sintomi legati al COVID;
- *terapia intensiva, totale ospedalizzati*, determina le persone gravi;
- *isolamento domiciliare*, determina le persone che sono state messe in isolamento dopo aver riscontrato il virus;
- *totale positivi*, determina il numero totale dei positivi nella giornata e la relativa *variazione totale dei positivi*;
- *nuovi positivi*, determina i nuovi positivi che si sono presentati nella giornata interessata;
- *dimessi guariti*, determina le persone guarite dal COVID-19;
- *deceduti*, determina le persone che purtroppo non ce l'hanno fatta;
- *totale casi*, determina il numero totale dei casi nella giornata;
- *numero tamponi*, determina il numero dei tamponi effettuati nella giornata.
- *casi testati*, determina il numero dei casi testati dopo l'attuazione del tampone.
- *possibili note in ita o in eng*.

Gli stessi elementi saranno poi usati per i dati delle **province** e quindi i relativi:

- *identificativi provincia e la relativa provincia e sigla*.

Il tutto a regime nell'arco temporale che parte dal 25 febbraio e termina il 3 Maggio.

Provvederemo dunque ad ulteriori informazioni utili all'analisi del fenomeno.

Per ogni **regione** indicheremo:

- *il numero di abitanti*,
- *la densità abitativa*,
- *la superficie in km²*,
- *numero residenti*,
- *numeri comuni*,
- *numero province*,
- *presidente*,
- *codice istat*,

- *codice fiscale,*
- *partiva iva,*
- *la pec,*
- *il sito web,*
- *la sede,*
- *numero degli aeroporti, stazioni, autostrade e strade statali.*

Per ogni **provincia** indicheremo:

- *il numero di abitanti,*
- *la densità abitativa,*
- *la superficie,*
- *numeri comuni,*
- *la sigla,*
- *latitudine*
- *longitudine*
- *codice e denominazione della regione appartenente*
- *superficie e densità di superficie*
- *residenti*
- *numero scuole e alberghi*

N.B la voce “in fase di aggiornamento/definizione” indica i possibili aggiornamenti o conferme che verranno accettate nei giorni a seguire.

e le relative **aziende sanitarie** presenti sul territorio indicando:

- *sigla provincia di residenza*
- *denominazione dell'azienda sanitaria*
- *codice dell'azienda*
- *numero di telefono*
- *fax*
- *email*
- *sito web*
- *partita iva*
- *codice della provincia di residenza*

Per ogni stato del mondo:

- *stato*
- *identificazione dello stato*
- *denominazione stato*
- *continente*
- *data casi dello stato*
- *numero casi*
- *deceduti*

Il DWH utilizza sorgenti di dati eterogenee provenienti da sistemi informativi esterni strutturati (**OLTP**) o da **flat files**, come nel caso in esame, ovvero si cercherà di aggiungere più informazioni possibili per dare un valore più significativo alle informazioni che la base di dati offrirà a chi interessato.

In uno scenario del genere, dunque, i dati delle varie sorgenti devono essere estratti opportunamente, ripuliti per eliminare eventuali incongruenze ed inconsistenze (Fase di Normalizzazione **Cap.3**), completati di parti mancanti ed integrati secondo uno schema comune (**Extraction, Transformation and Loading; ETL** dei dati). Dopo fatto ciò i dati riconciliati possono essere salvati in un'opportuna area di memoria detta **staging (Inflow)**.

2.1 Specifiche sulle operazioni

Le principali operazioni previste sulla base di dati sono:

Operazioni Data Marts

- Creazione di viste (**Cap.8**) che consentono agli utenti un accesso rapido alle informazioni più frequentemente usate migliorando i tempi di risposta del sistema. (**Outflow** dei dati)

Indicizzazioni

- Creazione di indici su tabelle più complesse per avere accesso alle informazioni in modo più efficiente e rapido. Un indice altro non è che una struttura dati che permette di organizzare in modo opportuno i record al fine di rendere efficiente appunto il recupero delle informazioni attraverso una chiave di ricerca sull'indice, che migliora la complessità dell'algoritmo di ricerca, che da lineare (esponenziale) diventa binaria (logaritmica).

Operazioni sulla base di dati (Upflow dei dati)

Operazioni di analisi ed estrazione dei dati dalla base dei dati

- Trend Settimanale;
- Variazione percentuale giornaliera;
- Tamponi giornalieri e contagiati;
- Le 5 regioni più colpite;
- Le 5 regioni meno colpite;
- Andamento delle province con più contagi;
- La classifica delle regioni con più terapie intensive e ricoveri;
- Numero di contagi per provincia in una determinata fascia temporale;
- Calcolo della percentuale contagi/tamponi;
- Calcolo della percentuale contagi/popolazioni italiana e sul mondo;
- Calcolo della letalità dell'epidemia;
- Calcolo dell'R0.
- Omissis...

Altre operazioni presenti nel capitolo delle query (Cap.5).

Creazione Trigger

Si farà uso dei **Trigger** per una appropriata consistenza dei dati ed una regolare gestione della base di dati con regole non esprimibili attraverso i costrutti dichiarativi del linguaggio SQL. Si prevedono, dunque, i seguenti trigger:

- Si segnalerà in caso di modifica dei dati un error (**exception**) dato che i dati possono essere solo caricati ed acceduti sulle seguenti tabelle, REGIONI_COVID19, PROVINCE_COVID19 e STATI_COVID19.
- Si calcolerà ad ogni inserimento dei dati:
 - La variazione percentuale dei casi rispetto al giorno precedente;
 - La variazione percentuale dei ricoveri in terapia intensiva rispetto al giorno precedente;
 - La variazione percentuale dei positivi rispetto al giorno precedente;
 - La variazione percentuale dei deceduti rispetto al giorno precedente;
 - La variazione percentuale dei tamponi effettuati rispetto al giorno precedente.

Per un'ulteriore analisi si rimanda il lettore al capitolo del livello dati (Cap.6)

2.2 Vincoli tecnologici e criteri di sicurezza dei dati

L'architettura prevista per l'applicazione di gestione dei dati sul contagio del COVID-19 è una classica architettura three-tier, ovvero realizzato a 3 livelli:

- **Livello di Presentazione** dove sarà presente l'interfaccia utente dell'applicazione web. Si è scelto l'applicazione web per un facile consulto dei dati relativi al coronavirus e i relativi grafici che renderanno l'idea al consultante, perché rappresenta un mezzo di comunicazione rapido ed efficace data l'elevato numero di utenti che oggi giorno navigano su internet. **(Cap.8)**
- **Livello Applicazione** ove saranno presenti gli oggetti software che realizzeranno una vera e propria logica applicativa, rappresenta un livello rivolto a coloro che oltre ad una mera conoscenza dei dati, hanno la necessità di prendere decisioni e quindi hanno bisogno anche di un'interpretazione dei dati sotto diversi punti di vista. **(Cap.7)**
- **Livello Dati** ove realizzeremo il DBMS e le informazioni da esso gestite, oltre a tutte quelle operazioni necessarie ai livelli superiori che sono logicamente indipendenti e non hanno bisogno di conoscere l'implementazione fisica (**information hiding**). **(Cap.6)**

Il DBMS sarà installato su un server dotato di 8 GB di RAM e un 1HD da 4TB di memoria di massa con possibilità di aggiornamento in seguito.

Una delle scelte migliori per l'archiviazione di informazioni, nel nostro caso, un **DWH** deve essere garantita la caratteristica di **affidabilità** e la **non volatilità**. Dunque sarebbe quella di adoperare una memoria stabile, ovvero non soggetta a danneggiamenti.

In particolare, ci si dovrebbe orientare verso un sistema **RAID**, introdotto nel 1988, che è composto da un insieme di hard disk che vengono visti dal sistema operativo come un normale singolo disco. Ciò permette attraverso un meccanismo del tutto nascosto all'utente di distribuire i dati su più dischi in maniera tale che il *Gestore dei Metodi di accesso e dei File* sia in grado di compiere più azioni contemporaneamente, migliorando le prestazioni generali del sistema. Ma cosa ancora più importante per l'**affidabilità** riguarda la possibilità di memorizzare le stesse informazioni su più dischi parallelamente, quindi, in caso di **guasto** le informazioni in esso memorizzate possono essere recuperate facilmente, senza problemi. A seconda di come vengono effettuate le due operazioni sopra enunciate, i sistemi RAID si differenziano in 7 livelli.

Gli utenti che dovranno interagire col sistema di basi di dati saranno:

- I **DBA** del sistema che possiederanno tutti i privilegi possibili sullo schema della base dei dati che possono visualizzare ed aggiornare le informazioni relative ai dati giornalieri del contagio.
- Gli **UTENTI**, che tramite un'applicazione web posso consultare tutti i dati del coronavirus tramite grafici e tabelle.

Capitolo 3

La progettazione concettuale

Lo scopo della progettazione concettuale è quello di rappresentare le specifiche informali della realtà di interesse in termini di una **descrizione formale e completa**, ma indipendente dai criteri di rappresentazione utilizzati nei sistemi di gestione di basi di dati (DBMS).

Il modello concettuale utilizzato è il modello Entità-Relazione (**Entity-Relationship ER**). In tale modello si prevede che i concetti ricavati dall'analisi dei requisiti vengano rappresentati da opportuni costrutti. I costrutti principali di tale modello sono:

Entità: rappresentano classi di oggetti che hanno proprietà comuni ed esistenza autonoma ai fini dell'applicazione di interesse. Una occorrenza di un'entità è un oggetto della classe che l'entità rappresenta.

Associazioni (Relazioni): rappresentano legami logici, significativi per l'applicazione di interesse, tra due o più entità. Una occorrenza di relazione è un'ennupla costituita da occorrenza di entità, una per ciascuna delle entità coinvolte.

Attributi: descrivono le proprietà elementari di entità o relazioni che sono di interesse ai fini dell'applicazione. Un attributo associa a ciascuna occorrenza di entità (o di relazione) un valore appartenente a un insieme, detto dominio, che contiene i valori ammissibili per l'attributo

Usando come riferimento lo schema della base di dati del **Cap.1** apportiamo modifiche necessarie affinché il sistema funzioni nei migliori dei modi **aggiungendo entità, attributi ed associazioni**.

3.1 Lo schema ER portante ed ER con cardinalità

Per prima cosa si estrae dalle specifiche ristrutturate lo schema ER portante con i concetti fondamentali.

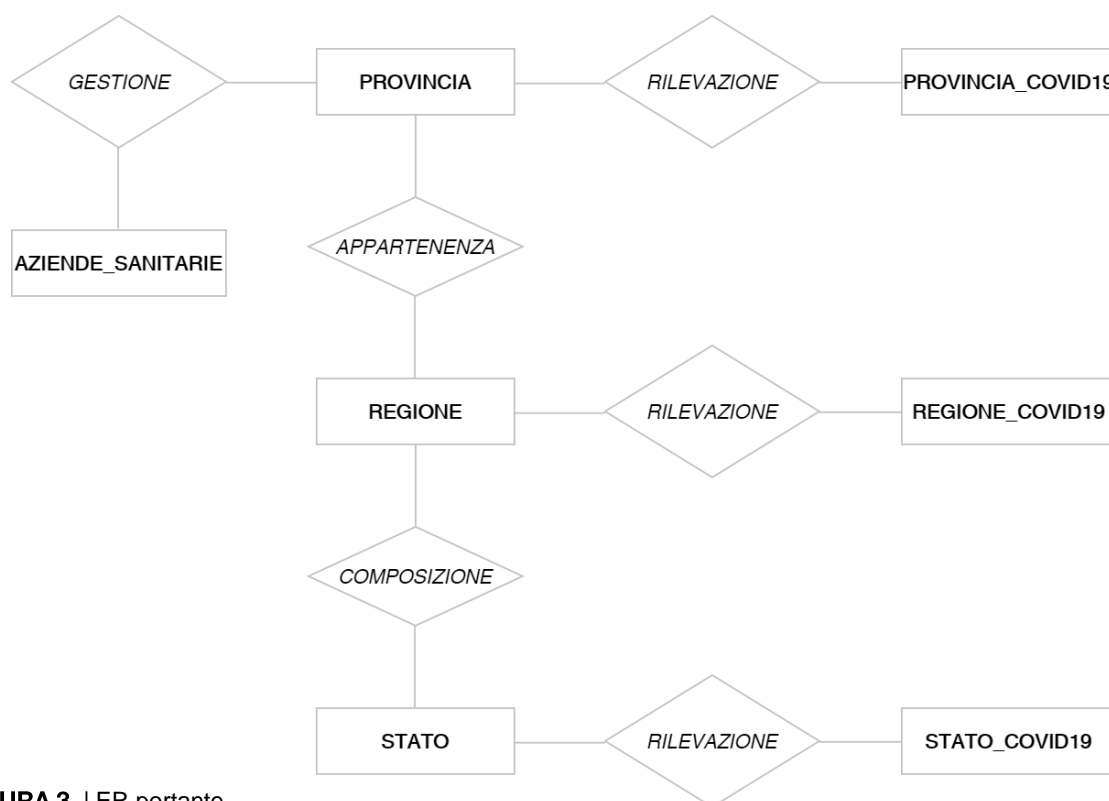


FIGURA 3 | ER portante

Dallo schema **ER portante** si evince subito l'aggiunta di nuove entità come Regione, Regione_Covid19, Stato, Stato_Covid19, Aziende Sanitarie. **(Figura 3)**

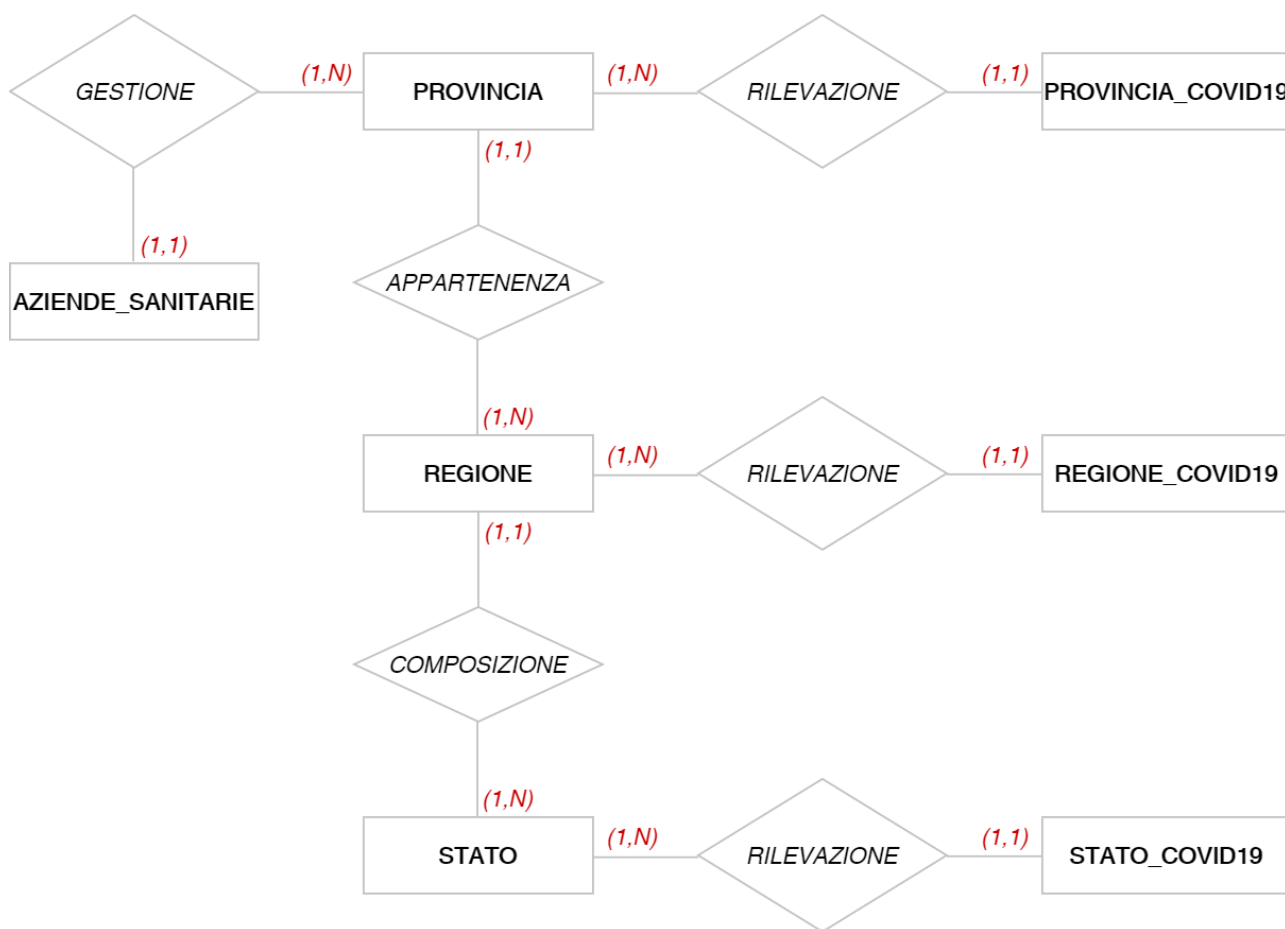


FIGURA 4 | ER con cardinalità

Dallo schema **ER con cardinalità (Figura 4)** indichiamo il numero minimo e massimo delle occorrenze di associazioni, cui ciascuna occorrenza di entità può partecipare. In questo caso avremo che:

- Ogni rilevazione nell'entità **STATO_COVID19** può riferirsi ad un solo stato, mentre ogni **STATO** presenta più rilevazioni in stato_covid19;
- Ogni **STATO** è composto da più regioni, mentre ogni **REGIONE** può comporre un solo stato;
- Ogni rilevazione nell'entità **REGIONE_COVID19** può riferirsi ad una sola regione, mentre ogni **REGIONE** presenta più rilevazioni in regione_covid19;
- Ogni **REGIONE** è composta da più province, mentre ogni **PROVINCIA** può comporre una sola regione;
- Ogni rilevazione nell'entità **PROVINCIA_COVID19** può riferirsi ad una sola provincia, mentre ogni **PROVINCIA** presenta più rilevazioni in provincia_covid19;
- Ogni **PROVINCIA** è gestita da più aziende sanitarie locali (ASL), mentre ogni **AZIENDA_SANITARIA** può gestire una sola provincia.

3.2 Glossario dei termini che verranno utilizzati nel database

La tabella elenca un glossario che verrà formalizzato nel database. Il nome campo indica le colonne della tabella della base dei dati, la relativa descrizione, il formato dei dati ed un piccolo esempio. Dividiamo il glossario tra le entità fondamentali della base dei dati.

Regioni

Nome campo	Descrizione	Formato	Esempio
<i>stato</i>	Stato di riferimento	XYZ (ISO 3166-1 alpha-3)	ITA
<i>codice_regione</i>	Codice della Regione (ISTAT 2019)	Numero	13
<i>denominazione_regione</i>	Denominazione della Regione	Testo	Abruzzo
<i>superficie</i>	Superficie della regione in km ²	Numero	43432.27
<i>densità_sup</i>	Densità abitativa per superficie della regione in km ²	Numero	535
<i>num_residenti</i>	Residenti nella regione	Numero	278438
<i>num_province</i>	Numero delle province appartenenti alla regione	Numero	6
<i>presidente</i>	Presidente della regione	Testo	Vincenzo De Luca
<i>cod_istat</i>	Il codice ISTAT della regione	Numero	15
<i>cod_fiscale</i>	Il codice fiscale della regione	Numero	80002870923
<i>piva</i>	La partita IVA della regione	Numero	481070423
<i>pec</i>	Indirizzo PEC della regione	Testo	dipartimento.presidenza@pec.regione.calabria.it
<i>sito</i>	Sito della regione	Testo	www.regione.calabria.it
<i>sede</i>	Sede della regione	Testo	Regione Calabria / Via Massara 2, 88100 Catanzaro
<i>num_aeroporti</i>	Aeroporti della regione	Numero	5
<i>num_stazioni</i>	Stazioni ferroviarie della regione	Numero	87
<i>num_autostrade</i>	Autostrade della regione	Numero	3
<i>num_strade_statali</i>	Strade statali della regione	Numero	49

Regioni_COVID19

Nome campo	Descrizione	Formato	Esempio
<i>codice_regione</i>	Codice della Regione (ISTAT 2019)	Numero	13
<i>denominazione_regione</i>	Denominazione della Regione	Testo	Abruzzo
<i>data</i>	Data dell'informazione	YYYY-MM-DD HH:MM:SS (ISO 8601) Ora italiana	2020-03-05 12:15:45

<i>ricoverati_con_sintomi</i>	Ricoverati con sintomi	Numero	3
<i>terapia_intensiva</i>	Ricoverati in terapia intensiva	Numero	3
<i>totale_ospedalizzati</i>	Totale ospedalizzati	Numero	3
<i>isolamento_domiciliare</i>	Persone in isolamento domiciliare	Numero	3
<i>totale_positivi</i>	Totale attualmente positivi (ospedalizzati + isolamento domiciliare)	Numero	3
<i>variazione_totale_positivi</i>	Variazione del totale positivi (totale_positivi giorno_corrente – totale_positivo giorno precedente)	Numero	3
<i>nuovi_positivi</i>	Nuovi attualmente positivi (totale_casi giorno_corrente - totale_casi giorno precedente)	Numero	3
<i>dimessi_guariti</i>	Persone dimesse guarite	Numero	3
<i>deceduti</i>	Persone decedute	Numero	3
<i>totale_casi</i>	Totale casi positivi	Numero	3
<i>tamponi</i>	Totale tamponi	Numero	3
<i>casi_testati</i>	Totale dei soggetti sottoposti al test	Numero	3
<i>note_it</i>	Note in lingua italiana (separate da ;)	Testo	pd-IT-000
<i>note_en</i>	Note in lingua inglese (separate da ;)	Testo	pd-EN-000

Province

Nome campo	Descrizione	Formato	Esempio
<i>codice_provincia</i>	Codice della Provincia (ISTAT 2019)	Numero	067
<i>denominazione_provincia</i>	Denominazione della provincia	Testo	Teramo
<i>sigla_provincia</i>	Sigla della Provincia	Testo	TE
<i>latitudine</i>	Latitudine	WGS84	42.6589177
<i>longitudine</i>	Longitudine	WGS84	13.70439971
<i>codice_regione</i>	Codice della Regione (ISTAT 2019)	Numero	13
<i>denominazione_regione</i>	Denominazione della Regione	Testo	Abruzzo
<i>superficie</i>	Superficie della provincia in km ²	Numero	54435.32
<i>densità_sup</i>	Densità abitativa per superficie della provincia in km ²	Numero	367
<i>residenti</i>	Residenti nella provincia	Numero	899974
<i>num_comuni</i>	Numero di comuni nella provincia	Numero	32
<i>num_scuole</i>	Numero di scuole	Numero	154

<i>num_alberghi</i>	Numero di alberghi	Numero	187
---------------------	--------------------	--------	-----

Province_COVID19

Nome campo	Descrizione	Formato	Esempio
<i>data_reg</i>	Data dell'informazione	YYYY-MM-DD HH:MM:SS (ISO 8601) Ora italiana	2020-03-05 12:15:45
<i>codice_provincia</i>	Codice della Provincia (ISTAT 2019)	Numero	067
<i>totale_casi</i>	Totale casi positivi	Numero	3
<i>note_it</i>	Note in lingua italiana (separate da ;)	Testo	pd-IT-000
<i>note_en</i>	Note in lingua inglese (separate da ;)	Testo	pd-EN-000

Stati

Nome campo	Descrizione	Formato	Esempio
<i>stato</i>	Stato di riferimento	XYZ (ISO 3166-1 alpha-3)	ITA
<i>descrizione</i>	Nome per esteso dello Stato	Testo	Italia
<i>capitale</i>	Capitale dello Stato	Testo	Roma
<i>popolazione</i>	Residenti nello Stato	Numero	67239220

Stati_COVID19

Nome campo	Descrizione	Formato	Esempio
<i>data_stato</i>	Data dell'informazione	YYYY-MM-DD HH:MM:SS (ISO 8601) Ora italiana	2020-03-05 12:15:45
<i>casi</i>	Casi del giorno nello Stato riferito	Numero	267
<i>deceduti</i>	Deceduti del giorno nello Stato riferito	Numero	86
<i>denominazioni_stato</i>	Denominazione dello Stato	Testo	Italia
<i>statoID</i>	Stato di riferimento compatto	ISO 3166-1 alpha-3	IT
<i>stato</i>	Stato di riferimento	ISO 3166-1 alpha-3	ITA
<i>continente</i>	Continente in cui si trova lo Stato	Testo	Europa

Aziende_Sanitarie

Nome campo	Descrizione	Formato	Esempio
<i>codice_azienza</i>	Codice dell'ASL	Numero	101
<i>denominazione_azienza</i>	Nome dell'ASL	Testo	ASL Napoli 3 Sud
<i>indirizzo</i>	Indirizzo dell'ASL	Testo	Corso Alcide de Gasperi, 7
<i>codice_provincia</i>	Codice della Provincia (ISTAT 2019)	Numero	067
<i>sigla_provincia</i>	Sigla della Provincia	Testo	TE
<i>telefono</i>	Contatto telefonico dell'ASL	Testo	0223837783
<i>fax</i>	Numero di fax dell'ASL	Testo	0223837783

email	E-mail dell'ASL	Testo	aslnapoli1centro@pec.a slna1centro.it
sito_web	Sito WEB dell'ASL	Testo	www.aslna1.napoli.it
partita_iva	Partita IVA dell'ASL	Numero	5841760829

3.3 Possibile raffinamento dello schema ER finale e le possibili trasformazioni o traduzioni

Se si vanno ora ad esaminare le specifiche ci si accorge che esistono ulteriori entità ed associazioni da prendere in considerazione.

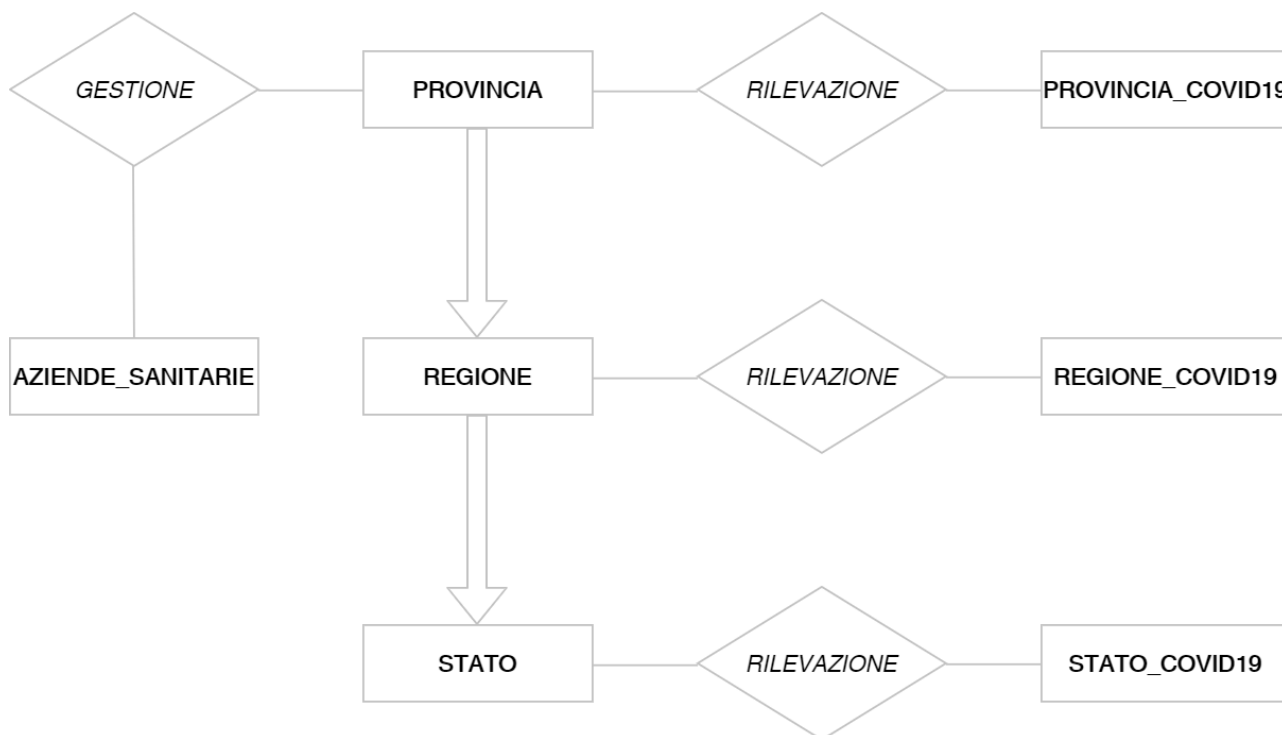


FIGURA 5 | ER avanzato, con generalizzazioni

Dallo schema precedente si nota come le entità tra loro possono avere in comune caratteristiche di tipo funzionali o strutturali, che ne consentono una classificazione strutturata **in una gerarchia di superclassi e sottoclassi (Figura 5)**.

Superclasse è una entità che accorpa sottoclassi distinte che hanno in comune alcune caratteristiche.

Tale tipo di associazione può essere anche detta relazione **is_a**. Nel caso in esame infatti avremo:

- **Provincia** è una **sottoclasse** di **Regione** (la provincia appartiene ad una determinata regione);
- **Regione** è una **sottoclasse** di **Stato** (la regione appartiene ad un determinato stato).

La gerarchia presente nel nostro caso è di tipo **(parziale, disgiunta)** ciò vuol dire che prese tutte le tuple della superclasse, accade che alcune di esse non siano presenti nella sottoclasse. Per esempio, nella tabella STATI può accadere che non tutti gli stati siano divisi per regioni, oppure ancora più evidente è il caso delle Province Autonome di Bolzano e Trento che sono considerate delle Regioni, ma che nella realtà non hanno province. In una situazione del genere, la soluzione più efficace, è quella di sostituire la gerarchia con una relazione (1,1) per la sottoclasse nei confronti della superclasse. Tale scelta è consigliabile anche perché sono previste operazioni diverse per ogni entità, inoltre, l'efficienza viene migliorata perché non si avranno campi NULL e non sarà necessario creare un attributo aggiuntivo per specificare il TIPO di entità.

3.4 ER finale della base di dati

Di seguito è riportato il diagramma **ER completo** per la base di dati in oggetto. **(Figura 6)**

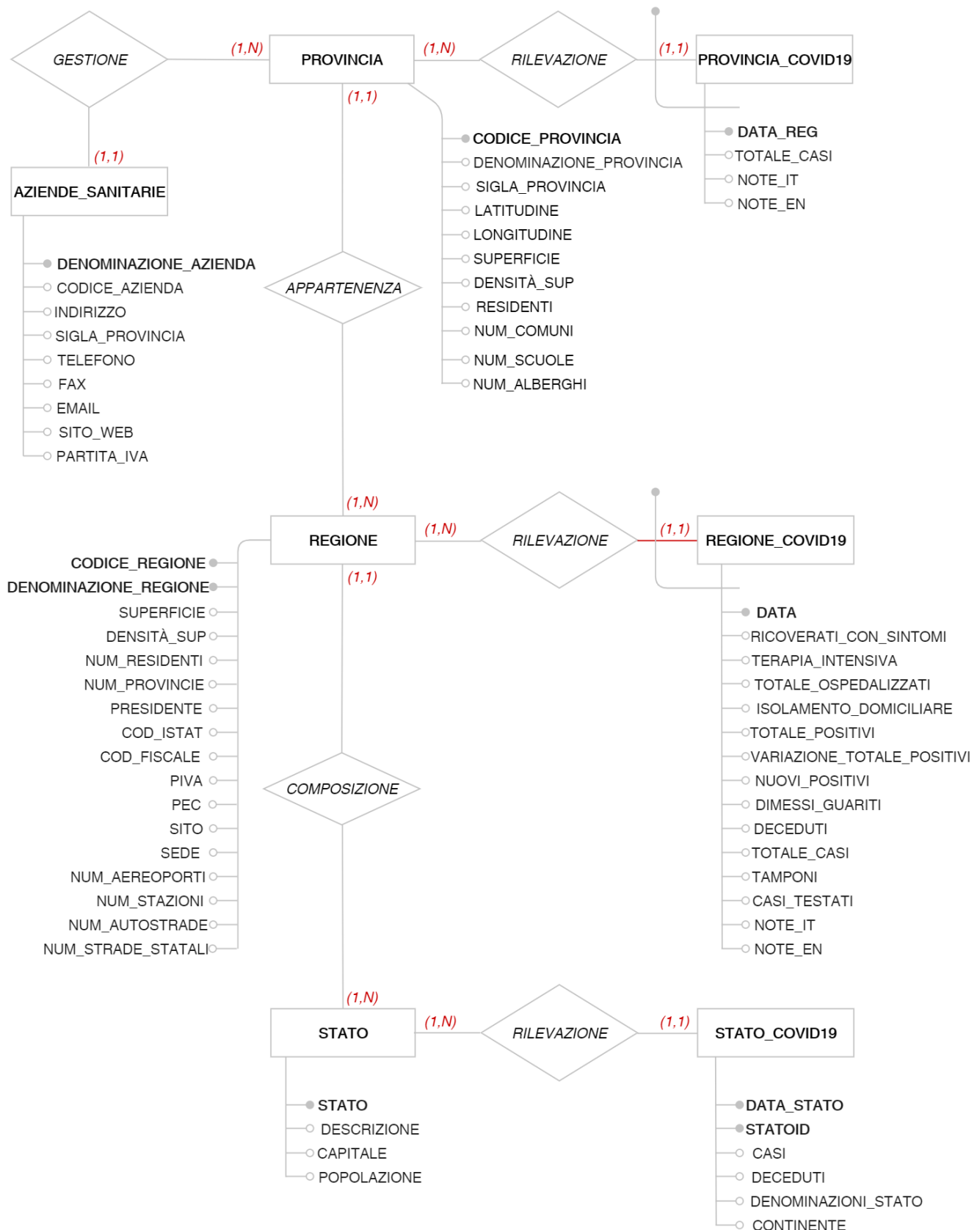


FIGURA 6 | ER finale

Dallo schema ER finale possiamo trarre alcune considerazioni molto importanti per l'analisi del fenomeno, infatti, innanzitutto notiamo come per ogni aggregazione territoriale siano rilevati i casi di Coronavirus, ciò indica l'importanza dei diversi punti di vista per mettere in condizione i vari enti territoriali di prendere decisioni sulle misure contenitive.

Nelle entità con il suffisso **_COVID19** sono presenti i dati temporali del fenomeno, mentre, nelle entità senza tale suffisso sono presenti le caratteristiche di ciascun territorio.

Questa distinzione che è stata fatta rappresenta un aspetto da non sottovalutare, infatti, avere il numero di casi di **Covid-19** in una **determinata regione**, può essere poco esplicativo, mentre se rapportato alla **densità abitativa** della regione potrebbe rendere tale dato più interessante (ad esempio, avere 500 casi in una regione con una densità abitativa di 1000km² è diverso da avere gli stessi casi in una regione con una densità abitativa di 2000km²).

Un ruolo diverso giocano le **AZIENDE_SANITARIE** esse rappresentano all'interno di questo database un supporto per coloro che volessero vedere nelle province con più casi, quali erano le Asl che dovevano gestire la situazione, anche per responsabilizzare tali enti ad un lavoro corretto.

D'altra parte è da notare inoltre come sono state attuate le trasformazioni e le traduzioni dallo schema ER avanzato.

Nell'ER Finale si sono indicati tutti gli attributi di ogni entità con identificazioni interne e siffatta analisi dall'ER avanzato si sono tradotte l'entità con identificazioni esterne.

Pertanto, realizzata la progettazione concettuale, si passa alla progettazione logica della base di dati. È possibile verificare lo schema delle relazioni alla **Figura 7 pag.21** ed è possibile notare i collegamenti tra le relazioni che indicano i vincoli di integrità referenziale.

Dopo aver dunque svolto la siffatta progettazione concettuale, si passa alla progettazione logica riconoscendo e costruendo uno schema orientato al modello relazionale ed in grafo di rappresentare gli stessi concetti modellati dallo schema concettuale.

In breve, la progettazione logica si suddivide in due particolari fasi:

- Fase di trasformazione, semplificando gli eventuali attributi composti o multivalore (nel nostro non sono presenti tali attributi);
- Fase di traduzione semplificando in un insieme di relazioni e di vincoli da rispettare.

È possibile inoltre verificare lo schema delle relazioni dopo un procedimento di **normalizzazione** nel **paragrafo 3.6**

3.5 Schema logico della base di dati

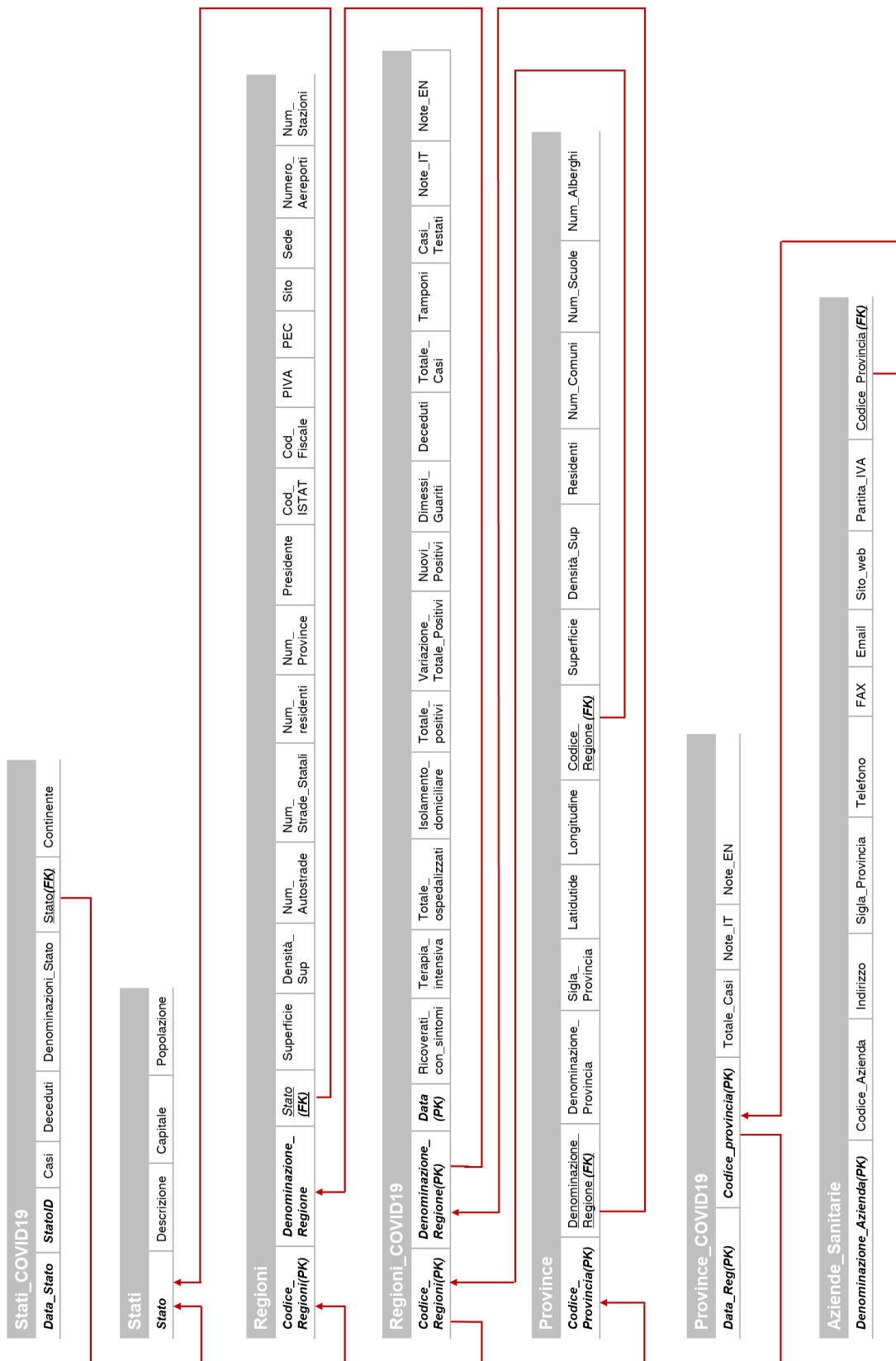


FIGURA 7 | Schema delle relazioni, progettazione logica.

3.6 Qualità dello schema logico e normalizzazione

La qualità dello schema logico viene valutata secondo gli strumenti messi a disposizione dalla teoria della normalizzazione.

La normalizzazione è un procedimento che consente di trasformare schemi non normalizzati in nuovi schemi i quali soddisfano una forma normale. Quando uno schema non soddisfa una forma normale esso presenta **ridondanza** ed **anomalie** nelle operazioni di aggiornamento, anomalie di cancellazione e anomalie di inserimento. Il processo di normalizzazione sottopone uno schema di relazione a una serie di test per "certificare" se soddisfa una data forma normale. Va detto che le metodologie di progetto viste nei capitoli precedenti permettono di solito di ottenere schemi che soddisfano una forma normale. Vi sono diverse forme normali, tutte soddisfatte dallo schema proposto. La prima forma normale (1NF) prevede che non vi siano attributi multivalore. La seconda e terza forma (2NF e 3NF) normale si occupano di verificare la bontà delle chiavi scelte, analizzando il rapporto che intercorre tra queste e gli attributi che da esse dipendono. Vi è un'altra forma normale, detta di Boyce e Codd.

Iniziamo a lavorare sul nostro schema relazionale **master**, cioè quello creato nel punto precedente (**Cap 1**). Lo **schema**:

MASTER (DATA_REG, STATO, CODICE_REGIONE, DENOMINAZIONE_REGIONE, CODICE_PROVINCIA, DENOMINAZIONE_PROVINCIA, SIGLA_PROVINCIA, LATITUDINE, LONGITUDINE, TOTALE_CASI, NOTE_IT, NOTE_EN).

Definiamo le dipendenze funzionali (DF), ovvero i legami che esistono tra gli attributi della relazione.

Questo passaggio ci aiuta a scoprire i vincoli d'integrità esistenti sullo schema di relazione, nel nostro caso abbiamo le **seguenti DF**:

- **DF0**: STATO → CODICE_REGIONE;
- **DF1**: CODICE_REGIONE → CODICE_PROVINCIA;
- **DF2**: CODICE_REGIONE → DENOMINAZIONE_REGIONE;
- **DF3**: CODICE_PROVINCIA → DENOMINAZIONE_PROVINCIA, SIGLA_PROVINCIA, LATITUDINE, LONGITUDINE.

(nota: Se indichiamo con $Y \rightarrow Z$, la dipendenza funzionale; Leggeremo che Y determina Z, o viceversa, che Z dipende funzionalmente da Y)

Adesso possiamo passare alla verifica della **3NF**, in particolare avremo che il nostro schema è in **3NF** se rispetta le seguenti caratteristiche:

- è in **2NF**;
- ogni attributo non primo di **MASTER** non dipende in modo transitivo dalla sua chiave, cioè dipende solo dalla chiave, e non vi sono dipendenze funzionali tra attributi non primi.

Il secondo punto non viene rispettato dalle **DF0** e **DF2**, in quanto tali dipendenze funzionali riguardano due attributi non primi. Quindi occorre fare due decomposizioni una sull'attributo **STATO** per la **DF0** ed un'altra sull'attributo **CODICE_REGIONE** per la **DF2**. Tali decomposizioni hanno lo scopo di mantenere le dipendenze funzionali, ma al contempo rispettare la **3NF** partendo quindi dal seguente **schema**:

MASTER (DATA_REG, STATO, CODICE_REGIONE, DENOMINAZIONE_REGIONE, CODICE_PROVINCIA, DENOMINAZIONE_PROVINCIA, SIGLA_PROVINCIA, LATITUDINE, LONGITUDINE, TOTALE_CASI, NOTE_IT, NOTE_EN).

Avremo la seguente decomposizione:

MASTER (DATA_REG, CODICE_REGIONE: REGIONI, CODICE_PROVINCIA,
DENOMINAZIONE_PROVINCIA, SIGLA_PROVINCIA, LATITUDINE, LONGITUDINE,
TOTALE_CASI, NOTE_IT, NOTE_EN);

REGIONI (CODICE_REGIONE, DENOMINAZIONE_REGIONE, STATO: STATI);

STATI (STATO).

Adesso ci rimane da verificare la **2NF**, ovvero dobbiamo controllare se il nostro schema soddisfa i seguenti requisiti:

- è in **1NF**;
- ogni attributo non primo dipende completamente da ogni chiave di R(X).

Anche in questo caso il secondo punto ci crea alcuni problemi, infatti, la **DF3** ci dice che alcuni attributi non primi non dipendono completamente dalla chiave dello schema, ma solo da un sottoinsieme di essa.

Quando uno schema non soddisfa la **2NF** come in questo caso, la soluzione migliore è quella di decomporre lo schema, tuttavia, tale operazione va fatta in maniera corretta al fine di evitare problemi nella consistenza dei dati e nel rispetto dei vincoli.

In particolare, ciò a cui dobbiamo ambire in questa fase è una **decomposizione senza perdita**, per ottenere ciò bisogna effettuare la decomposizione sulla base di ‘*attributi comuni*’ che contengono una chiave per almeno una delle relazioni decomposte. Quindi, nel nostro caso avremo che il seguente schema:

MASTER (DATA_REG, CODICE_REGIONE: REGIONI, CODICE_PROVINCIA,
DENOMINAZIONE_PROVINCIA, SIGLA_PROVINCIA, LATITUDINE, LONGITUDINE,
TOTALE_CASI, NOTE_IT, NOTE_EN);

REGIONI (CODICE_REGIONE, DENOMINAZIONE_REGIONE, STATO: STATI);

STATI (STATO).

Viene **decomposto** in questa maniera:

MASTER (DATA_REG, CODICE_REGIONE: REGIONI, CODICE_PROVINCIA: PROVINCE,
TOTALE_CASI, NOTE_IT, NOTE_EN);

PROVINCE (CODICE_PROVINCIA, DENOMINAZIONE_PROVINCIA, SIGLA_PROVINCIA,
LATITUDINE, LONGITUDINE);

REGIONI (CODICE_REGIONE, DENOMINAZIONE_REGIONE, STATO: STATI);

STATI (STATO).

Infine, la verifica della **1NF** è **banale** in quanto ogni attributo di **MASTER** è un **attributo semplice**, ovvero con dominio atomico.

In definitiva il nostro schema di relazione adesso è in **3NF**, tuttavia, la **DF1** non rispetta la forma normale di **Boyce** e **Codd** perché abbiamo che un attributo non primo **determina** un attributo primo. Per eliminare tale problema, basta creare un’associazione tra le entità PROVINCE e REGIONI, in altri termini occorre fare una decomposizione su CODICE_PROVINCIA, avremo quindi lo schema finale:

MASTER (DATA_REG, CODICE_PROVINCIA: PROVINCE, TOTALE_CASI, NOTE_IT,
NOTE_EN);

PROVINCE (CODICE_PROVINCIA, CODICE_REGIONE: REGIONI,
DENOMINAZIONE_PROVINCIA, SIGLA_PROVINCIA, LATITUDINE, LONGITUDINE);

REGIONI (CODICE_REGIONE, DENOMINAZIONE_REGIONE, STATO: STATI);

STATI (STATO).

Se dunque lo schema è nella Boyce&Codd Normal Form allora lo schema è anche in terza forma normale.

3.6.1 - Comandi ddl

--Creo la tabella STATI con lo stesso tipo che era definito nella tabella MASTER

```
CREATE TABLE STATI (
    STATO VARCHAR2(3) not null,
    PRIMARY KEY (STATO)
);
```

--Inserisco nella tabella STATI i valori (non duplicati) che prima erano presenti nella tabella MASTER
INSERT INTO STATI(STATO) SELECT DISTINCT STATO FROM MASTER;

Output su Datagrip

	STATO
1	ITA

--Creo la tabella REGIONI con lo stesso tipo che era definito nella tabella MASTER

```
CREATE TABLE REGIONI (
    CODICE_REGIONE NUMBER not null,
    DENOMINAZIONE_REGIONE VARCHAR2(21) not null,
    STATO VARCHAR2(3) not null,
    FOREIGN KEY (STATO) REFERENCES STATI(STATO) ON DELETE CASCADE,
    PRIMARY KEY (CODICE_REGIONE, DENOMINAZIONE_REGIONE)
);
```

--Inserisco nella tabella REGIONI i valori (non duplicati) che prima erano presenti nella tabella MASTER

INSERT INTO REGIONI(CODICE_REGIONE, DENOMINAZIONE_REGIONE, STATO) SELECT DISTINCT CODICE_REGIONE, DENOMINAZIONE_REGIONE, STATO FROM MASTER;

--Output su Datagrip (per questioni di comodità vengono mostrate solo le prime 10 tuple)

	CODICE_PROVINCIA	DENOMINAZIONE_PROVINCIA	SIGLA_PROVINCIA	LATITUDINE	LONGITUDINE	CODICE_REGIONE	DENOMINAZIONE_REGIONE
1	5	Asti	AT	44.89912921	8.204142547	1	Piemonte
2	96	Biella	BI	45.56651120	8.054082167	1	Piemonte
3	3	Novara	NO	45.44508506	8.621915884	1	Piemonte
4	98	Sassari	SS	40.72667657	8.559667131	20	Sardegna
5	993	In fase di definizione/aggiornamento	<null>	0.00000000	0.00000000	20	Sardegna
6	48	Firenze	FI	43.76923077	11.255888850	9	Toscana
7	46	Lucca	LU	43.84432283	10.501513660	9	Toscana
8	995	In fase di definizione/aggiornamento	<null>	0.00000000	0.00000000	9	Toscana
9	26	Treviso	TV	45.66754571	12.245073630	5	Veneto
10	23	Verona	VR	45.43839046	10.993526850	5	Veneto

	 CODICE_REGIONE	 DENOMINAZIONE_REGIONE	 STATO
1	1	Piemonte	ITA
2	19	Sicilia	ITA
3	2	Valle d'Aosta	ITA
4	8	Emilia-Romagna	ITA
5	7	Liguria	ITA
6	14	Molise	ITA
7	17	Basilicata	ITA
8	15	Campania	ITA
9	6	Friuli Venezia Giulia	ITA
10	16	Puglia	ITA

--Creo la tabella REGIONI con lo stesso tipo che era definito nella tabella MASTER

```
CREATE TABLE PROVINCE (
    CODICE_PROVINCIA          NUMBER not null,
    DENOMINAZIONE_PROVINCIA  VARCHAR2(36) not null,
    SIGLA_PROVINCIA          VARCHAR2(2),
    LATITUDINE                NUMBER(11, 8) not null,
    LONGITUDINE               NUMBER(11, 9) not null,
    CODICE_REGIONE            NUMBER not null,
    DENOMINAZIONE_REGIONE    VARCHAR2(21) not null,
    FOREIGN KEY (CODICE_REGIONE, DENOMINAZIONE_REGIONE) REFERENCES
    REGIONI(CODICE_REGIONE, DENOMINAZIONE_REGIONE) ON DELETE CASCADE,
    PRIMARY KEY (CODICE_PROVINCIA)
```

);

--Inserisco nella tabella REGIONI i valori (non duplicati) che prima erano presenti nella tabella MASTER

```
INSERT INTO PROVINCE (CODICE_PROVINCIA, DENOMINAZIONE_PROVINCIA,
SIGLA_PROVINCIA, LATITUDINE, LONGITUDINE, CODICE_REGIONE,
DENOMINAZIONE_REGIONE)
```

```
SELECT DISTINCT CODICE_PROVINCIA, DENOMINAZIONE_PROVINCIA, SIGLA_PROVINCIA,
LATITUDINE, LONGITUDINE, CODICE_REGIONE, DENOMINAZIONE_REGIONE FROM
MASTER;
```

--Output su Datagrip (per questioni di comodità vengono mostrate solo le prime 10 tuple)

	 CODICE_PROVINCIA	 DENOMINAZIONE_PROVINCIA	 SIGLA_PROVINCIA	 LATITUDINE	 LONGITUDINE	 CODICE_REGIONE	 DENOMINAZIONE_REGIONE
1	5	Asti	AT	44.89912921	8.284142547	1	Piemonte
2	96	Biella	BI	45.56651128	8.054082167	1	Piemonte
3	3	Novara	NO	45.44588586	8.621915884	1	Piemonte
4	98	Sassari	SS	48.72667657	8.559667131	20	Sardegna
5	993	In fase di definizione/aggiornamento	<null>	0.00000000	0.00000000	20	Sardegna
6	48	Firenze	FI	43.76923077	11.255888858	9	Toscana
7	46	Lucca	LU	43.84432283	10.501513660	9	Toscana
8	995	In fase di definizione/aggiornamento	<null>	0.00000000	0.00000000	9	Toscana
9	26	Treviso	TV	45.66754571	12.245073630	5	Veneto
10	23	Verona	VR	45.43839046	10.993526850	5	Veneto

--Adesso possiamo modificare la tabella MASTER ed eliminare gli attributi denormalizzati ed aggiungere la Foreign Key di PROVINCE

```
ALTER TABLE MASTER DROP COLUMN STATO;
ALTER TABLE MASTER DROP COLUMN CODICE_REGIONE;
ALTER TABLE MASTER DROP COLUMN DENOMINAZIONE_REGIONE;
ALTER TABLE MASTER DROP COLUMN DENOMINAZIONE_PROVINCIA;

ALTER TABLE MASTER DROP COLUMN SIGLA_PROVINCIA;
ALTER TABLE MASTER DROP COLUMN LATITUDINE;
ALTER TABLE MASTER DROP COLUMN LONGITUDINE;
ALTER TABLE MASTER ADD CONSTRAINT MASTER_PROVINCE_FK FOREIGN KEY
(CODICE_PROVINCIA) REFERENCES PROVINCE(CODICE_PROVINCIA) ON DELETE
CASCADE;
ALTER TABLE MASTER RENAME TO PROVINCE_COVID19;
```

--Output su Datagrip (per questioni di comodità vengono mostrate solo le prime 10 tuple)

	DATA_REG	CODICE_PROVINCIA	TOTALE_CASI	NOTE_IT	NOTE_EN
1	2020-02-26T18:00:00	109	0	<null>	<null>
2	2020-02-26T18:00:00	43	0	<null>	<null>
3	2020-02-26T18:00:00	41	1	<null>	<null>
4	2020-02-26T18:00:00	989	0	<null>	<null>
5	2020-02-26T18:00:00	70	0	<null>	<null>
6	2020-02-26T18:00:00	94	0	<null>	<null>
7	2020-02-26T18:00:00	990	0	<null>	<null>
8	2020-02-26T18:00:00	6	0	<null>	<null>
9	2020-02-26T18:00:00	5	0	<null>	<null>
10	2020-02-26T18:00:00	96	0	<null>	<null>

Capitolo 4

La progettazione fisica

La progettazione fisica di una base di dati è un processo complesso che prevede l'attuazione delle seguenti fasi:

- Dimensionamento fisico della base di dati, in termini di calcolo dello storage (spazio di memorizzazione) richiesto su disco;
- Creazione del database;
- Definizione delle politiche di sicurezza e creazione degli utenti/ruoli
- Creazione degli oggetti della base di dati e definizione dei vincoli;
- Creazione di un'istanza (popolamento della base di dati).

4.1 Condizioni operative del database

Qui si elenca tutto quello che è possibile utilizzare nella realizzazione del database

- Installazione della licenza di ORACLE 18XE o altro;
- Installato su Server con 8CPU 4,7 GHz, 8 GB di RAM, 1HD da 4TB e sistema RAID
- Sistema operativo del server: Windows Server 2019

4.2 Dimensionamento fisico della base di dati

Stesura di una stima di costi in termini di occupazione di memoria della base di dati per la successiva fase di implementazione.

Occupazione in termini di byte:

TIPO	PESO BYTE
Number(x)	$[x/2]+2$ byte
Date	7 byte
Char (x)	X byte
Varchar(x)	0...x byte

Di seguito viene riportata per ogni tabella una stima dell'occupazione di memoria a regime.

Regioni

Attributo	Tipo	Byte	Storage (B)
Codice_Regione	Number(2)	3	63
Denominazione_Regione	Varchar(21)	21	441
Stato	Varchar(3)	3	63
Superficie	Number(8)	6	126
Densità_sup	Number(3)	4	84
Num_residenti	Number(8)	6	126
Num_Province	Number(2)	3	63
Presidente	Varchar(100)	100	2100
Cod_Istat	Number(2)	3	63

Cod_Fiscale	Number(11)	8	168
PIVA	Number(11)	8	168
PEC	Varchar(100)	100	2100
Sito	Varchar(100)	100	2100
Sede	Varchar(100)	100	2100
Num_aereoporti	Number(2)	3	63
Num_stazioni	Number(3)	4	84
Num_autostrade	Number(2)	3	63
Num_Strade_Statali	Number(2)	3	63
Totale Storage			9802KB

Regioni_COVID19

Attributo	Tipo	Byte	Storage(B)
Data	Varchar(19)	19	27531
Codice_Regione	Number(2)	3	4347
Denominazione_Regione	Varchar(21)	21	30429
Ricoverati_Con_Sintomi	Number(5)	5	7245
Terapia_Intensiva	Number(4)	4	5.796
Totale_Ospedalizzati	Number(5)	5	7245
Isolamento_Domiciliare	Number(5)	5	7245
Totale_Positivi	Number(5)	5	7245
Variazione_Totale_Positivi	Number(4)	4	5.796
Nuovi_Positivi	Number(4)	4	5.796
Dimessi_Guariti	Number(5)	5	7245
Deceduti	Number(5)	5	7245
Totale_Casi	Number(5)	5	7245
Tamponi	Number(6)	5	7245
Casi_testati	Number(4)	4	5.796
Note_IT	Varchar2(10)	10	14490
Note_EN	Varchar2(10)	10	14490
Totale Storage			16839KB

Province

Attributo	Tipo	Byte	Storage(B)
Codice_Provincia	Number(3)	4	512
Denominazione_Provincia	Varchar(36)	36	4608
Denominazione_Regione	Varchar(21)	21	2688
Sigla_Provincia	Varchar(2)	2	256
Latitudine	Number(11)	7	896
Longitudine	Number(11)	7	896
Codice_Regione	Number(2)	3	384
Superficie	Number(7)	5	640
Densità_sup	Number(4)	4	512
Residenti	Number(7)	6	768
Num_Comuni	Number(3)	4	512
Num_Scuole	Number(4)	4	512
Num_Alberghi	Number(4)	4	512
Totale Storage			13375KB

Province_COVID19

Attributo	Tipo	Byte	Storage(B)
Data_reg	Varchar(19)	19	167808
Codice_Provincia	Number(3)	4	35328
Totale_casi	Number(5)	5	44160
Note_IT	Varchar2(10)	10	88.320
Note_EN	Varchar2(10)	10	88.320
Totale Storage			414KB

Aziende_Sanitarie

Attributo	Tipo	Byte	Storage(B)
Codice_Provincia	Number(3)	4	396
Codice_Azienda	Number(3)	4	396
Indirizzo	Varchar(39)	39	3.861
Sigla_Provincia	Varchar(2)	2	198
Telefono	Varchar(12)	12	1.188
Fax	Varchar(12)	12	1.188
Email	Varchar(43)	43	4.257
Sito_Web	Varchar(27)	27	2.673
Partita_Iva	Number(11)	8	792
Totale Storage			14598KB

Stati

Attributo	Tipo	Byte	Storage(B)
Stato	Varchar(3)	3	615
Descrizione	Varchar(40)	40	8200
Capitale	Varchar(150)	150	30750
Popolazione	Number(10)	7	1435
Totale Storage			40039KB

Stati_COVID19

Attributo	Tipo	Byte	Storage(B)
Data_stato	Varchar(10)	10	146580
Casi	Number(5)	5	73290
Deceduti	Number(4)	4	58632
Denominazioni_Stato	Varchar(42)	42	615636
StatoID	Varchar(8)	8	117264
Stato	Varchar(3)	3	43974
Continente	Varchar(7)	7	102606
Totale Storage			52963KB

4.3 Configurazione del DB server e dimensionamento totale della base di dati

Prima di procedere all'implementazione e modifica fisica vera e propria della base di dati è necessario effettuare alcune importanti operazioni ai fini di una corretta gestione dello spazio di memoria disponibile. Tali operazioni consistono in un partizionamento della memoria (disco) in più blocchi di grandezza differente ognuno adibito al contenimento di un determinato gruppo di file accomunati da analoghe caratteristiche

Oggetto	Denominazione_Tabella	Storage in byte
Tabella	Regioni	9802KB
Tabella	Regioni_Covid19	16839KB
Tabella	Province	13375KB
Tabella	Province_Covid19	414KB
Tabella	Stati	40039KB
Tabella	Azienda_Sanitarie	14598KB
Tabella	Stati_Covid19	52963KB
TOTALE STORAGE		14456MB → 14.456 GB

Il partizionamento pensato sapendo che il **DBMS** è **ORACLE**, sarà del seguente tipo:

- C: [Dimensione 30 GB] – Partizione per Software di Base (OS)
- D: [Dimensione 30 GB] – Partizione dedicata all'installazione del DBMS ORACLE e dei tool di gestione
- E: [Dimensione 10 GB] – Partizione dedicata ai Control File di ORACLE
- F: [Dimensione 150 GB] – Partizione dedicata ai LogFile di ORACLE
- G: [Dimensione 1 TB] – Partizione dedicata ai DataFile di ORACLE

Capitolo 5

Creazione della base di dati

Dopo aver normalizzato la tabella master, passiamo alla creazione del database.

La fase di definizione è il punto cruciale della realizzazione delle relazioni in un base di dati, si definiscono infatti le relazioni: **STATI**, **STATI_COVID19**, **PROVINCE**, **PROVINCE_COVID_19**, **REGIONI**, **REGIONI_COVID_19**, **AZIENDE_SANITARIE** attuando i vincoli di integrazioni intra-relazionale e inter-relazionali.

Innanzitutto viene creato una **tablespace**, ovvero si crea un'area fisica di memorizzazione dove le tabelle che andremo a creare saranno memorizzate, di seguito il codice **SQL** relativo:

```
create tablespace DB_COVID19 datafile
'G:/tablespace/DB_COVID19.dbf' SIZE 50GB
```

Sulla base dei calcoli effettuati nel capitolo precedente ove il dimensionamento totale della base di dati prevista è di ~14GB e dato che la base di dati è progettata per uno sviluppo futuro magari per i prossimi 3, max 4 mesi di ciclo di terminazione del virus (**Cap.6 per verificare l'analisi**).

Con riferimento alle specifiche sulle politiche di sicurezza, si è deciso di creare un utente DBA (*Database Administrator*) ovvero chi gestisce la base di dati e un ruolo utente standard che può accedere ai dati senza ne caricarli e ne modificarli. Di seguito il codice **SQL** relativo:

```
create user DBA_COVID19 default tablespace DB_COVID19 identified by Administrator
GRANT dba, unlimited tablespace TO DBA_COVID19
```

```
create role USER_COVID19
GRANT CONNECT ON DB_COVID19.* TO USER_COVID19
GRANT SELECT ON DB_COVID19.* TO USER_COVID19
```

Da notare come l'utente covid19 ha il solo privilegio di selezionare gli elementi nella base di dati proprio per evitare anomalie di **non volatilità** del DWH in esame.

Di seguito la creazione degli oggetti della base di dati e i relativi **create table**:

PROVINCE

```
create table PROVINCE
(
  CODICE_PROVINCIA      NUMBER      not null
    primary key,
  DENOMINAZIONE_PROVINCIA VARCHAR2(36) not null,
  SIGLA_PROVINCIA       VARCHAR2(2),
  LATITUDINE            NUMBER(11, 8) not null,
  LONGITUDINE           NUMBER(11, 9) not null,
  CODICE_REGIONE        NUMBER      not null,
  DENOMINAZIONE_REGIONE VARCHAR2(21) not null,
  SUPERFICIE            NUMBER(7, 2),
  DENSITÀ_SUP           NUMBER,
  RESIDENTI             NUMBER,
  NUM_COMUNI            NUMBER,
```

```

NUM_SCUOLE          NUMBER,
NUM_ALBERGHI        NUMBER,
foreign key (CODICE_REGIONE, DENOMINAZIONE_REGIONE) references REGIONI
);

```

PROVINCE_COVID19

create table PROVINCE_COVID19

```

(
  DATA_REG          VARCHAR2(19) not null,
  CODICE_PROVINCIA    NUMBER      not null
    constraint MASTER_PROVINCE_FK
    references PROVINCE
    on delete cascade,
  TOTALE_CASI         NUMBER      not null,
  NOTE_IT             VARCHAR2(10),
  NOTE_EN             VARCHAR2(10),
  primary key (DATA_REG, CODICE_PROVINCIA)
);

```

REGIONI

create table REGIONI

```

(
  CODICE_REGIONE      NUMBER      not null,
  DENOMINAZIONE_REGIONE VARCHAR2(21) not null,
  STATO               VARCHAR2(3)
    references STATI
    on delete set null,

  SUPERFICIE          NUMBER(8, 2),
  DENSITÀ_SUP         NUMBER,
  NUM_RESIDENTI        NUMBER,
  NUM_PROVINCIE        NUMBER,
  PRESIDENTE          VARCHAR2(100),
  COD_ISTAT            NUMBER,
  COD_FISCALE          NUMBER,
  PIVA                 NUMBER,
  PEC                  VARCHAR2(100),
  SITO                 VARCHAR2(100),
  SEDE                 VARCHAR2(100),
  NUM_AEREOPORTI       NUMBER,
  NUM_STAZIONI         NUMBER,
  NUM_AUTOSTRADE       NUMBER,
  NUM_STRADE_STATALI   NUMBER,
  constraint REGIONI_PK
  primary key (CODICE_REGIONE, DENOMINAZIONE_REGIONE)
);

```

REGIONI_COVID19

create table REGIONI_COVID19

```

(
  DATA               VARCHAR2(19) not null,

```

```

CODICE_REGIONE          NUMBER    not null,
DENOMINAZIONE_REGIONE   VARCHAR2(21) not null,
RICOVERATI_CON_SINTOMI   NUMBER,
TERAPIA_INTENSIVA        NUMBER,
TOTALE_OSPEDALIZZATI     NUMBER,
ISOLAMENTO_DOMICILIARE   NUMBER,
TOTALE_POSITIVI          NUMBER,
VARIAZIONE_TOTALE_POSITIVI NUMBER,
NUOVI_POSITIVI           NUMBER,
DIMESSI_GUARITI          NUMBER,
DECEDUTI                 NUMBER,
TOTALE_CASI              NUMBER,
TAMPONI                  NUMBER,
CASI_TESTATI             NUMBER,
NOTE_IT                  VARCHAR2(10),
NOTE_EN                  VARCHAR2(10),
primary key (DATA, CODICE_REGIONE, DENOMINAZIONE_REGIONE),
foreign key (CODICE_REGIONE, DENOMINAZIONE_REGIONE) references REGIONI
    on delete cascade
);

```

STATI

create table STATI

```

(
    STATO          VARCHAR2(3) not null
        primary key,
    DESCRIZIONE     VARCHAR2(40),
    POPOLAZIONE     NUMBER,
    CAPITALE        VARCHAR2(150)
);

```

STATI_COVID19

create table STATI_COVID19

```

(
    DATA_STATO      VARCHAR2(10) not null,
    CASI              NUMBER    not null,
    DECEDUTI          NUMBER    not null,
    DENOMINAZIONI_STATO VARCHAR2(42) not null,
    STATOID           VARCHAR2(8) not null,
    STATO            VARCHAR2(3)
        constraint STATI_COVID19_STATI_FK
            references STATI,
    CONTINENTE        VARCHAR2(7) not null,
    primary key (DATA_STATO, STATOID)
);

```

AZIENDE_SANITARIE

create table AZIENDE_SANITARIE

```

(

```

```

CODICE_AZIENDA          NUMBER    not null,
DENOMINAZIONE_AZIENDA   VARCHAR2(40) not null
    primary key,
INDIRIZZO               VARCHAR2(39) not null,
SIGLA_PROVINCIA         VARCHAR2(2)  not null,
TELEFONO                VARCHAR2(12),
FAX                     VARCHAR2(12),
EMAIL                   VARCHAR2(43),
SITO_WEB                VARCHAR2(27),
PARTITA_IVA             NUMBER,
CODICE_PROVINCIA        NUMBER
    constraint AZIENDE_SANITARIE_PROVINCE_FK
    references PROVINCE
);

```

È possibile visualizzare e provare le seguenti definizioni in SQL, oltre al **riempimento completo** della base di dati e gli eventuali **alter table** [Cliccando qui](#).
Oppure al seguente link <https://paste.ee/r/tfIFR/0>

5.1 Query and Reporting

Il Query and Reporting è il processo che permette di porre interrogazioni, recuperare dati fondamentali dalla base di dati. È un processo per trasformare i dati in modo appropriato e porre i risultati in formato leggibile verso gli utenti attraverso grafici e diagrammi.

Analizzando il fenomeno, si è dunque pensato di realizzare le seguenti query e reporting e in alcune query si è opportunamente diagrammato.

5.1.1 Andamento completo dell'epidemia, totale ospedalizzati, totale positivi, totale casi, totale deceduti, totale dimessi guariti;

Query

```

SELECT DATA, sum(TOTALE_OSPEDALIZZATI) AS TOT_OSP, sum(TOTALE_POSITIVI) AS
Tot_pos, sum(TOTALE_CASI) AS Tot_cas, sum(DECEDUTI) AS Tot_deceduti,
sum(DIMESSI_GUARITI) AS Tot_dim
FROM REGIONI_COVID19
GROUP BY DATA
ORDER BY DATA ASC ;

```

Tabella

	DATA	TOT_OSP	TOT_POS	TOT_CAS	TOT_DECEDUTI	TOT_DIM
1	2020-02-25T18:00:00	149	311	322	10	1
2	2020-02-26T18:00:00	164	385	400	12	3
3	2020-02-27T18:00:00	304	588	650	17	45
4	2020-02-28T18:00:00	409	821	888	21	46
5	2020-02-29T17:00:00	506	1049	1128	29	50
6	2020-03-01T17:00:00	779	1577	1694	34	83
7	2020-03-02T18:00:00	908	1835	2036	52	149
8	2020-03-03T18:00:00	1263	2263	2502	79	160
9	2020-03-04T17:00:00	1641	2706	3089	107	276
10	2020-03-05T17:00:00	2141	3296	3858	148	414
11	2020-03-06T17:00:00	2856	3916	4636	197	523
12	2020-03-07T18:00:00	3218	5061	5883	233	589
13	2020-03-08T18:00:00	4207	6387	7375	366	622
14	2020-03-09T18:00:00	5049	7985	9172	463	724
15	2020-03-10T18:00:00	5915	8514	10149	631	1004
16	2020-03-11T17:00:00	6866	10590	12462	827	1045
17	2020-03-12T17:00:00	7803	12839	15113	1016	1258
18	2020-03-13T17:00:00	8754	14955	17660	1266	1439
19	2020-03-14T17:00:00	9890	17750	21157	1441	1966
20	2020-03-15T17:00:00	11335	20603	24747	1809	2335
21	2020-03-16T17:00:00	12876	23073	27980	2158	2749
22	2020-03-17T17:00:00	14954	26062	31506	2503	2941
23	2020-03-18T17:00:00	16620	28710	35713	2978	4025
24	2020-03-19T17:00:00	18255	33190	41035	3405	4440
25	2020-03-20T17:00:00	18675	37860	47021	4032	5129
26	2020-03-21T17:00:00	20565	42681	53578	4825	6072
27	2020-03-22T17:00:00	22855	46638	59138	5476	7024
28	2020-03-23T17:00:00	23896	50418	63927	6077	7432
29	2020-03-24T17:00:00	25333	54030	69176	6820	8326
30	2020-03-25T17:00:00	26601	57521	74386	7503	9362

La tabella continua fino al 3 Maggio

5.1.2 Contagi per provincia in un determinato arco temporale;

Query

```
SELECT DENOMINAZIONE_PROVINCIA, sum(TOTALE_CASI) AS Totale_Casi
FROM PROVINCE_COVID19 P JOIN PROVINCE P2 ON P.CODICE_PROVINCIA =
P2.CODICE_PROVINCIA
WHERE data_REG <= '2020-03-25T17:00:00' AND data_REG >= '2020-03-19T17:00:00' AND
DENOMINAZIONE_PROVINCIA <> 'In fase di definizione/aggiornamento'
GROUP BY DENOMINAZIONE_PROVINCIA
ORDER BY TOTALE_CASI DESC ;
```

Tabella

	DENOMINAZIONE_PROVINCIA	TOTALE_CASI
1	Bergamo	42155
2	Brescia	38040
3	Milano	33951
4	Cremona	19448
5	Torino	13774
6	Piacenza	12449
7	Lodi	12151
8	Pavia	9137
9	Padova	8853
10	Pesaro e Urbino	8632
11	Parma	8395
12	Reggio nell'Emilia	7917
13	Monza e della Brianza	7674
14	Modena	7388
15	Roma	7261
16	Verona	7101
17	Rimini	6471
18	Treviso	6364
19	Mantova	6360
20	Trento	6256
21	Lecco	5921
22	Bologna	5209
23	Alessandria	5092
24	Venezia	4957
25	Bolzano	4646
26	Ancona	4473
27	Genova	4420
28	Vicenza	4404
29	Como	3604
30	Firenze	3594

La tabella continua...

5.1.3 Regione con il maggior numero di casi per densità abitativa;

Query

```

SELECT * FROM ( SELECT R.DENOMINAZIONE_REGIONE, sum(R.TOTALE_CASI) AS
TOTALE_CASI, R2.DENSITÀ_SUP AS DENSITÀ_ABITATIVA_IN_KM2
FROM REGIONI_COVID19 R JOIN REGIONI R2 ON R.CODICE_REGIONE =
R2.CODICE_REGIONE and R.DENOMINAZIONE_REGIONE =
R2.DENOMINAZIONE_REGIONE
GROUP BY R.DENOMINAZIONE_REGIONE, R2.DENSITÀ_SUP
ORDER BY R2.DENSITÀ_SUP, TOTALE_CASI DESC )
WHERE ROWNUM = 1;

```


Tabella

	DENOMINAZIONE_REGIONE	TOTALE_CASI	DENSITÀ_ABITATIVA_IN_KM2
1	Basilicata	12580	56

5.1.4 Tamponi giornalieri e contagiati;
Query

```
SELECT DENOMINAZIONE_REGIONE, TAMPONI, TOTALE_POSITIVI
FROM REGIONI_COVID19
WHERE DATA ='2020-05-03T17:00:00' ORDER BY TOTALE_POSITIVI DESC ;
```

Tabella

	DENOMINAZIONE_REGIONE	TAMPONI	TOTALE_POSITIVI
1	Lombardia	410857	36926
2	Piemonte	172208	15638
3	Emilia-Romagna	197075	9045
4	Veneto	378202	7299
5	Toscana	150914	5328
6	Lazio	150912	4385
7	Liguria	54492	3551
8	Marche	64412	3198
9	Puglia	66443	2955
10	Campania	86498	2726
11	Sicilia	85955	2203
12	Abruzzo	40699	1868
13	P.A. Trento	41095	1247
14	Friuli Venezia Giulia	74990	1087
15	Calabria	38835	702
16	Sardegna	27737	689
17	P.A. Bolzano	44240	665
18	Basilicata	14210	194
19	Umbria	38823	183
20	Molise	7075	181
21	Valle d'Aosta	8100	109

5.1.5 Le 5 Regioni con più decessi fino al 3 Maggio;
Query

```
SELECT *
FROM ( SELECT DENOMINAZIONE_REGIONE, DECEDUTI
      FROM REGIONI_COVID19
      WHERE DATA ='2020-05-03T17:00:00' ORDER BY DECEDUTI DESC )
WHERE ROWNUM <= 5 ;
```

Tabella

	 DENOMINAZIONE_REGIONE	 DECEDUTI
1	Lombardia	14231
2	Emilia-Romagna	3642
3	Piemonte	3152
4	Veneto	1516
5	Liguria	1209



5.1.6 Le 5 Regioni con meno decessi fino al 3 Maggio;

Query

```

SELECT *
FROM ( SELECT DENOMINAZIONE_REGIONE, DECEDUTI
      FROM REGIONI_COVID19
      WHERE DATA ='2020-05-03T17:00:00' ORDER BY DECEDUTI ASC )
WHERE ROWNUM <= 5 ;
    
```

Tabella

	 DENOMINAZIONE_REGIONE	 DECEDUTI
1	Molise	22
2	Basilicata	25
3	Umbria	68
4	Calabria	88
5	Sardegna	119

5.1.7 Le 10 Province con più casi fino al 3 Maggio + Grafico;

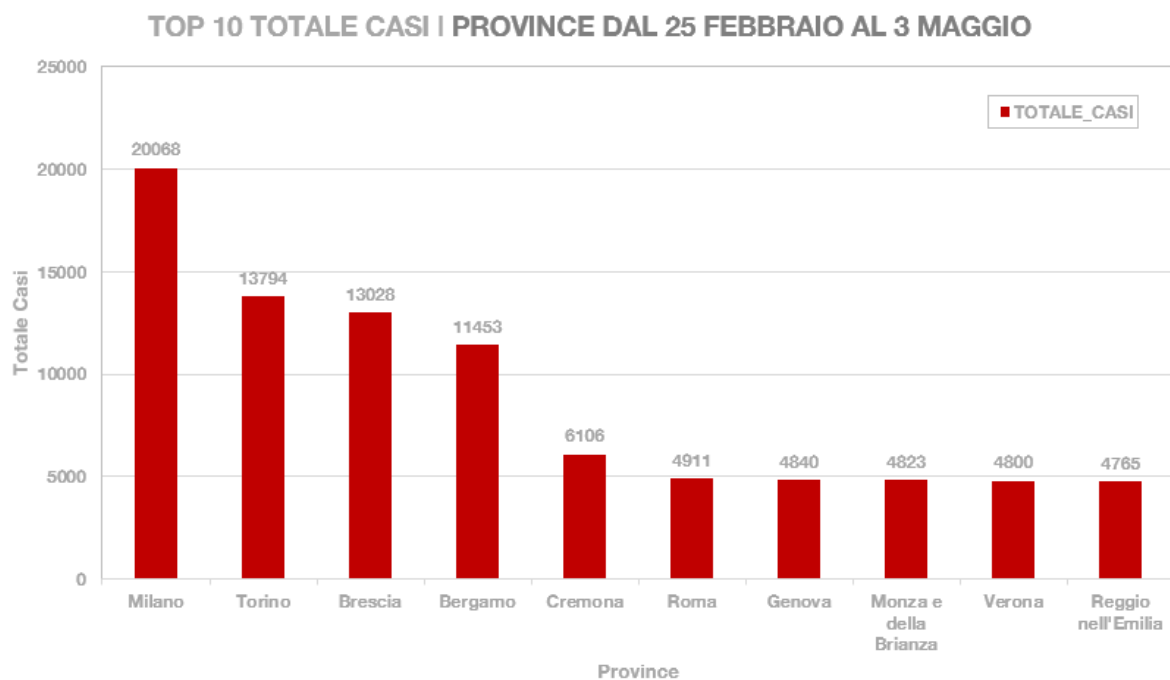
Query

```
SELECT *
FROM ( SELECT DENOMINAZIONE_PROVINCIA, R.DENOMINAZIONE_REGIONE,
TOTALE_CASI
      FROM (PROVINCE_COVID19 P JOIN PROVINCE P2 ON P.CODICE_PROVINCIA =
P2.CODICE_PROVINCIA) JOIN REGIONI R ON P2.CODICE_REGIONE =
R.CODICE_REGIONE
      WHERE DATA_REG ='2020-05-03T17:00:00' ORDER BY TOTALE_CASI DESC )
WHERE ROWNUM <= 10 ;
```

Tabella

	DENOMINAZIONE_PROVINCIA	DENOMINAZIONE_REGIONE	TOTALE_CASI
1	Milano	Lombardia	20068
2	Torino	Piemonte	13794
3	Brescia	Lombardia	13028
4	Bergamo	Lombardia	11453
5	Cremona	Lombardia	6106
6	Roma	Lazio	4911
7	Genova	Liguria	4840
8	Monza e della Brianza	Lombardia	4823
9	Verona	Veneto	4800
10	Reggio nell'Emilia	Emilia-Romagna	4765

Grafico



5.1.8 Le 10 Province con meno casi fino al 3 Maggio + Grafico;

Query

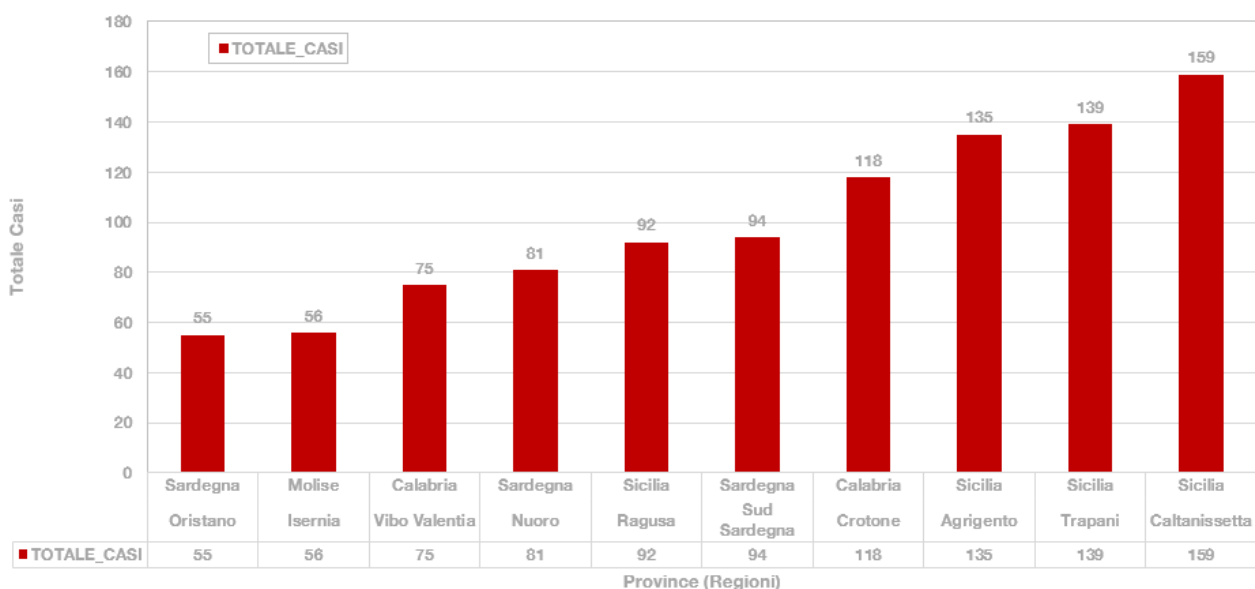
```
SELECT * FROM ( SELECT DENOMINAZIONE_PROVINCIA, R.DENOMINAZIONE_REGIONE,
TOTALE_CASI
FROM (PROVINCE_COVID19 P JOIN PROVINCE P2 ON P.CODICE_PROVINCIA =
P2.CODICE_PROVINCIA) JOIN REGIONI R ON P2.CODICE_REGIONE =
R.CODICE_REGIONE
WHERE DATA_REG ='2020-05-03T17:00:00' AND DENOMINAZIONE_PROVINCIA
<> 'In fase di definizione/aggiornamento'
ORDER BY TOTALE_CASI ASC )
WHERE ROWNUM <= 10 ;
```

Tabella

	DENOMINAZIONE_PROVINCIA	DENOMINAZIONE_REGIONE	TOTALE_CASI
1	Oristano	Sardegna	55
2	Isernia	Molise	56
3	Vibo Valentia	Calabria	75
4	Nuoro	Sardegna	81
5	Ragusa	Sicilia	92
6	Sud Sardegna	Sardegna	94
7	Crotone	Calabria	118
8	Agrigento	Sicilia	135
9	Trapani	Sicilia	139
10	Caltanissetta	Sicilia	159

Grafico

TOP 10 TOTALE CASI | Province meno colpite






5.1.9 Andamento terapia intensiva, ricoverati con sintomi per regioni;

Query

```
SELECT DENOMINAZIONE_REGIONE, TERAPIA_INTENSIVA, RICOVERATI_CON_SINTOMI
FROM REGIONI_COVID19
WHERE DATA ='2020-05-03T17:00:00'
ORDER BY TERAPIA_INTENSIVA DESC ;
```

Tabella

	 DENOMINAZIONE_REGIONE ÷	 TERAPIA_INTENSIVA ÷	 RICOVERATI_CON_SINTOMI ÷
1	Lombardia	532	6609
2	Emilia-Romagna	197	1997
3	Piemonte	169	2496
4	Toscana	112	513
5	Veneto	103	955
6	Lazio	95	1346
7	Liguria	68	627
8	Marche	43	400
9	Puglia	40	410
10	Campania	30	455
11	Sicilia	29	383
12	P.A. Trento	17	136
13	Abruzzo	16	300
14	Umbria	13	58
15	P.A. Bolzano	11	109
16	Sardegna	10	92
17	Friuli Venezia Giulia	6	131
18	Calabria	4	95
19	Basilicata	3	48
20	Valle d'Aosta	2	74
21	Molise	1	8

5.2.0 Contagi nel mondo dal 1 gennaio al 3 Maggio;**Query**

```
SELECT DENOMINAZIONI_STATO, sum(CASI) AS TOTALE_CASI
FROM STATI_COVID19
GROUP BY DENOMINAZIONI_STATO
ORDER BY TOTALE_CASI DESC ;
```

Tabella

	DENOMINAZIONI_STATO	TOTALE_CASI
1	United_States_of_America	1133069
2	Spain	216582
3	Italy	209328
4	United_Kingdom	182260
5	Germany	162496
6	France	130979
7	Turkey	124375
8	Russia	124054
9	Brazil	96559
10	Iran	96448
11	China	83961
12	Canada	56714
13	Belgium	49517
14	Peru	42534
15	Netherlands	40236
16	India	39980
17	Switzerland	29734
18	Ecuador	27464
19	Saudi_Arabia	25459
20	Portugal	25190
21	Mexico	22088
22	Sweden	22082
23	Ireland	21176
24	Pakistan	19103
25	Chile	18435
26	Singapore	17548
27	Israel	16185
28	Belarus	15828
29	Austria	15558
30	Qatar	14872

La tabella continua...

5.2.1 Contagi nei continenti dal 1 Gennaio al 3 Maggio;

Query

```
SELECT CONTINENTE, sum(CASI) AS TOTALE_CASI
FROM STATI_COVID19
GROUP BY CONTINENTE
ORDER BY TOTALE_CASI DESC ;
```

Tabella

	CONTINENTE	TOTALE_CASI
1	America	1434136
2	Europe	1361853
3	Asia	541019
4	Africa	42778
5	Oceania	8183
6	Other	696

Nelle query successivi si è realizzato una visualizzazione grafica dei risultati ottenuti dalle interrogazioni poste sulla base di dati.

Di seguito (**Figura 8**) uno schema che riassume i procedimenti che si sono seguiti per la realizzazione del grafico.

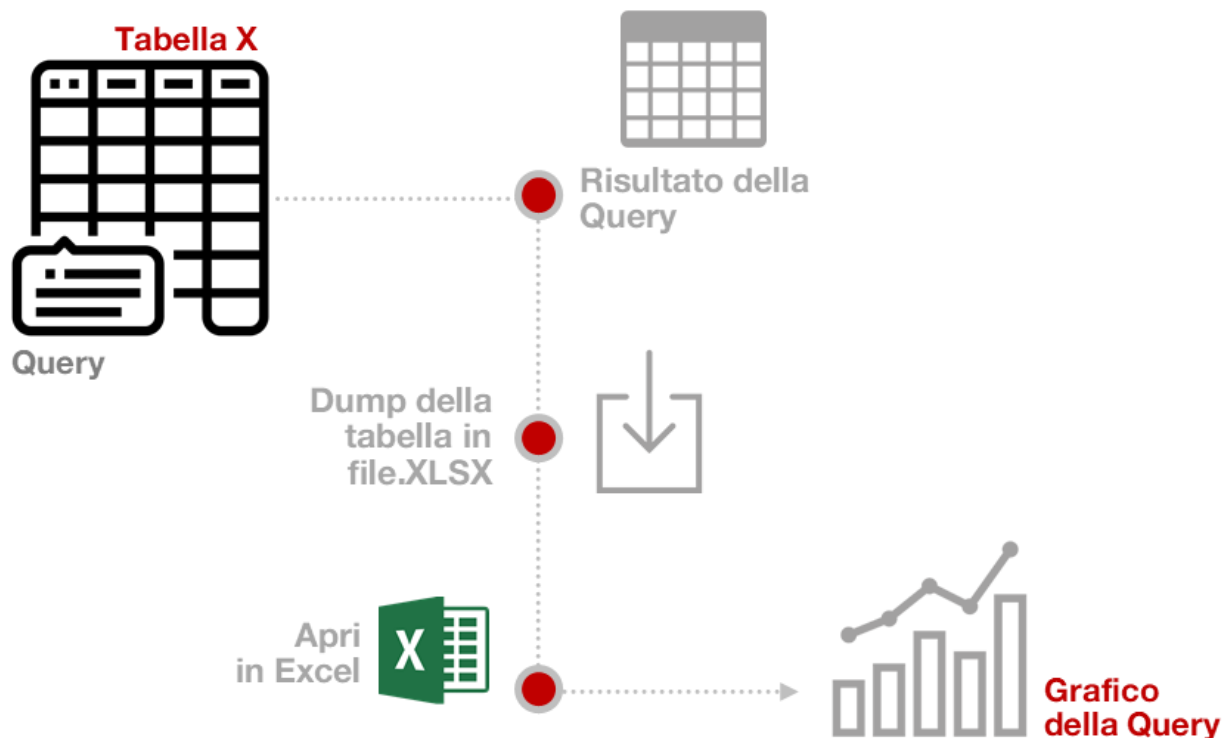


FIGURA 8 | Procedura di realizzazione grafici delle query.

5.2.2 I primi 10 stati con più casi dal 1 gennaio al 3 Maggio + Grafico;

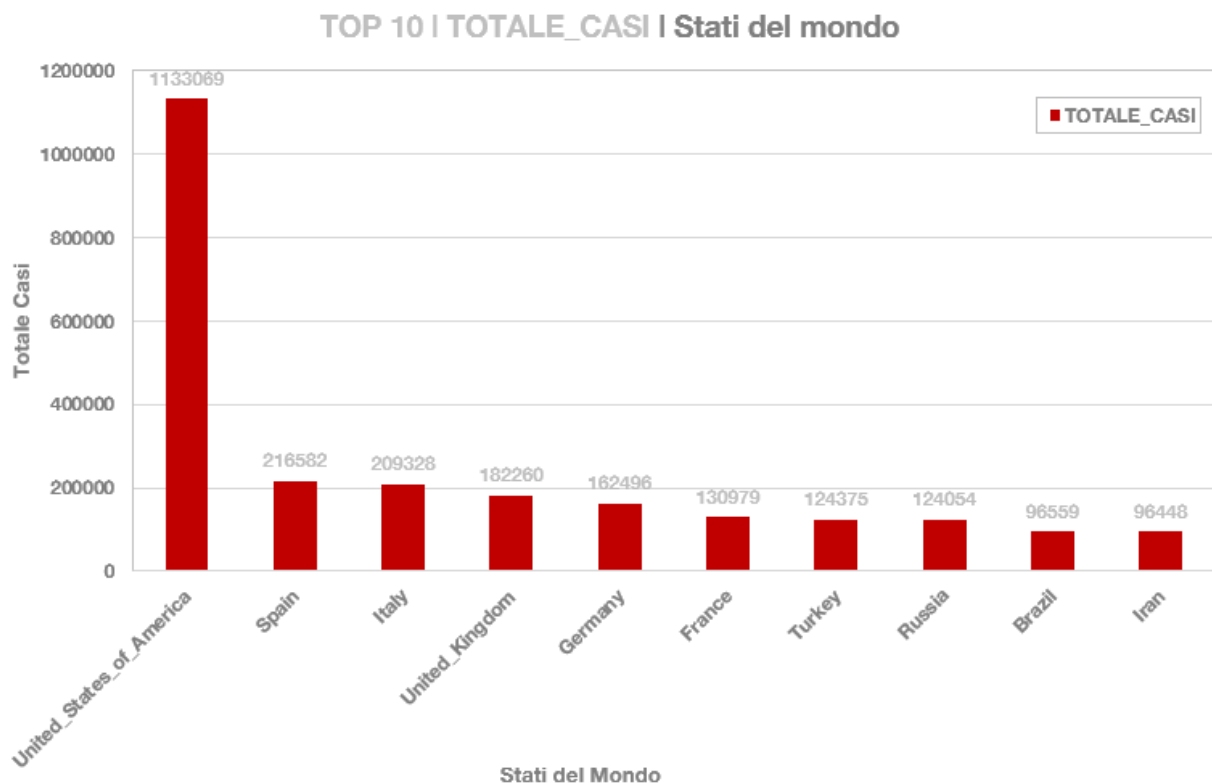
Query

```
SELECT * FROM ( SELECT DENOMINAZIONI_STATO, sum(CASI) AS TOTALE_CASI
FROM STATI_COVID19
GROUP BY DENOMINAZIONI_STATO
ORDER BY TOTALE_CASI DESC )
WHERE ROWNUM <= 10 ;
```

Tabella

	DENOMINAZIONI_STATO	TOTALE_CASI
1	United_States_of_America	1133069
2	Spain	216582
3	Italy	209328
4	United_Kingdom	182260
5	Germany	162496
6	France	130979
7	Turkey	124375
8	Russia	124054
9	Brazil	96559
10	Iran	96448

Grafico



5.2.3 Classifica dei continenti con più casi + Grafico.

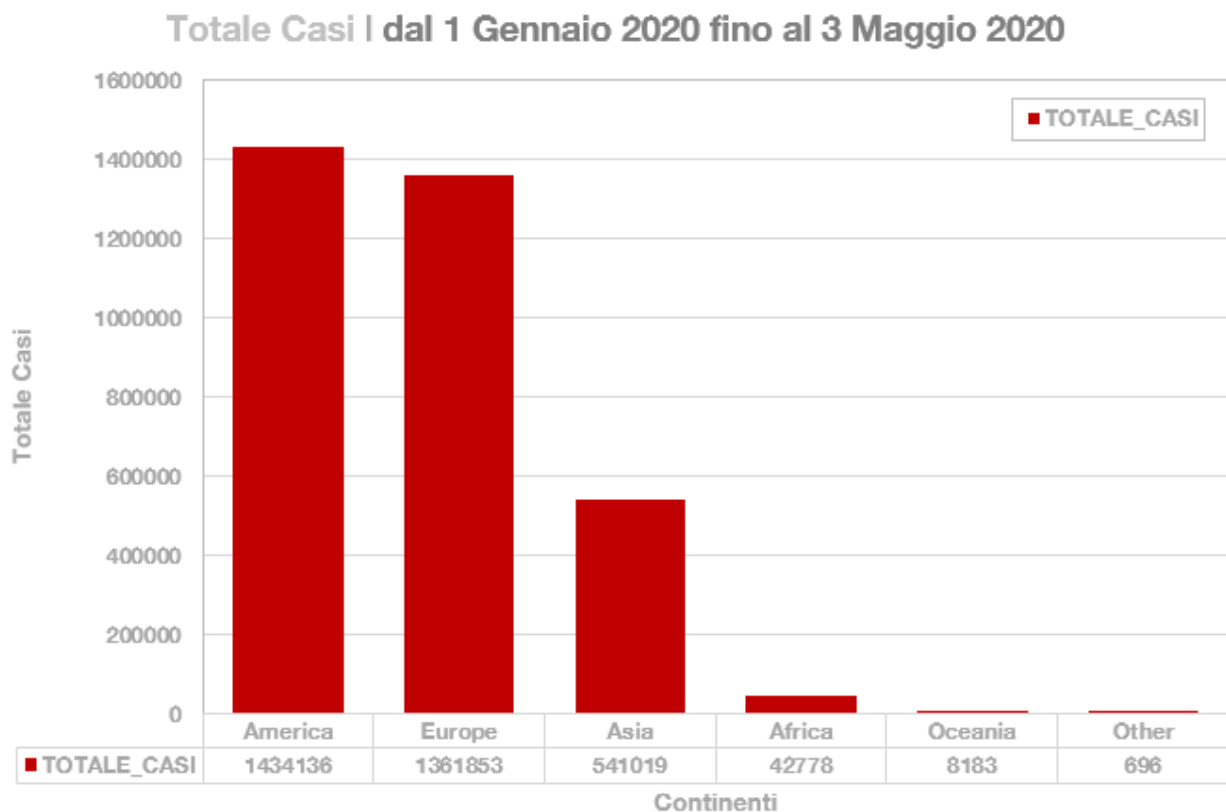
Query

```
SELECT CONTINENTE, sum(CASI) AS TOTALE_CASI
FROM STATI_COVID19
GROUP BY CONTINENTE
ORDER BY TOTALE_CASI DESC ;
```

Tabella

	CONTINENTE	TOTALE_CASI
1	America	1434136
2	Europe	1361853
3	Asia	541019
4	Africa	42778
5	Oceania	8183
6	Other	696

Grafico



È possibile visualizzare e provare le seguenti query in SQL [Cliccando qui](#)
Oppure al seguente link <https://paste.ee/r/JzPyR/0>

5.2 Guida alla lettura dei dati attraverso query

Va premesso che questi dati con ogni probabilità sottostimano il numero di contagi (e, ormai è chiaro, anche di decessi), perché alcune persone - pur avendo i sintomi di Covid-19 - non vengono sottoposte a tampone, e dunque «sfuggono» al conteggio.

Ma i dati del database sono importanti in ogni caso: perché indicano una tendenza e, soprattutto, perché ci danno con estrema precisione lo stato di saturazione degli ospedali.

Questa è una **guida** che tenta di spiegare come leggere i dati del nostro database, e soprattutto come estrarre correttamente da lì le risposte ad alcune domande chiave. Ad esempio, quante persone, ufficialmente, si sono contagiate, nelle 24 ore, in un determinato giorno in Italia? Ed il numero di contagiati è salito o no, e di quanto?

Altrettanto importante è capire a quali domande può rispondere il nostro database, nonostante le apparenze: ad esempio, quante persone sono state ricoverate in terapia intensiva, in Italia, da un giorno all' altro? O quante sono guarite?

Compilato su base regionale prendiamo, ad esempio, il documento relativo al 25 marzo.

Query

```
SELECT *
FROM REGIONI_COVID19
WHERE DATA ='2020-03-25T17:00:00'
ORDER BY DENOMINAZIONE_REGIONE ASC ;
```

Tabella

	DATA	CODICE ...	DENOMINAZIONE_R...	RICOVERATI...	TERAPIA_INTENSIVA	TOTALE OSPEDALIZZATI	ISOLAMENTO DOMICILIARE	TOTALE_POSITIVI	
1	2020-03-25T17:00:00	13	Abruzzo	248	59	307	431	738	
2	2020-03-25T17:00:00	17	Basilicata	20	14	34	78	112	
3	2020-03-25T17:00:00	18	Calabria	93	23	116	217	333	
4	2020-03-25T17:00:00	15	Campania	318	123	441	631	1072	
5	2020-03-25T17:00:00	8	Emilia-Romagna	3180	294	3474	4782	8256	
6	2020-03-25T17:00:00	6	Friuli Venezia Giulia	200	52	252	659	911	
7	2020-03-25T17:00:00	12	Lazio	805	101	906	769	1675	
8	2020-03-25T17:00:00	7	Liguria	927	147	1074	752	1826	
9	2020-03-25T17:00:00	3	Lombardia	10026	1236	11262	9329	20591	
10	2020-03-25T17:00:00	11	Marche	938	148	1086	1553	2639	
11	2020-03-25T17:00:00	14	Molise	26	7	33	20	53	
12	2020-03-25T17:00:00	4	P.A. Bolzano	190	40	230	518	748	
13	2020-03-25T17:00:00	4	P.A. Trento	308	65	373	685	1058	
14	2020-03-25T17:00:00	1	Piemonte	2544	381	2925	2631	5556	
15	2020-03-25T17:00:00	16	Puglia	349	64	413	610	1023	
16	2020-03-25T17:00:00	20	Sardegna	82	19	101	311	412	
17	2020-03-25T17:00:00	19	Sicilia	259	80	339	597	936	
18	2020-03-25T17:00:00	9	Toscana	999	251	1250	1526	2776	
19	2020-03-25T17:00:00	10	Umbria	123	44	167	519	686	
20	2020-03-25T17:00:00	2	Valle d'Aosta	70	25	95	280	375	
21	2020-03-25T17:00:00	5	Veneto	1407	316	1723	4022	5745	
	VARIAZIONE_TOTALE_POSITIVI	NUOVI_POSITIVI	DIMESSI_GUARITI	DECEDUTI	TOTALE_CASI	TAMPONI	CASI_TESTATI	NOTE_IT	NOTE_EN
1	116	124	23	52	813	4982	<null>	<null>	<null>
2	21	21	0	1	113	857	<null>	<null>	<null>
3	29	32	7	11	351	5058	<null>	<null>	<null>
4	80	98	53	74	1199	6972	<null>	<null>	<null>
5	545	800	721	1077	10054	38045	<null>	<null>	<null>
6	63	147	158	70	1139	9494	<null>	<null>	<null>
7	130	173	131	95	1901	20669	<null>	<null>	<null>
8	134	189	225	254	2305	6602	<null>	<null>	<null>
9	723	1643	7281	4474	32346	81666	<null>	<null>	<null>
10	142	198	8	287	2934	7896	<null>	<null>	<null>
11	-2	0	12	8	73	580	<null>	<null>	<null>
12	49	77	67	43	858	6649	<null>	<null>	<null>
13	83	112	90	74	1222	4114	<null>	<null>	<null>
14	432	509	19	449	6024	16655	<null>	<null>	<null>
15	83	88	22	48	1093	8223	<null>	<null>	<null>
16	17	21	12	18	442	3019	<null>	<null>	<null>
17	137	148	33	25	994	8312	<null>	<null>	<null>
18	257	273	54	142	2972	17868	<null>	<null>	<null>
19	62	62	5	19	710	4707	<null>	<null>	<null>
20	-4	1	2	24	401	1200	<null>	<null>	<null>
21	394	494	439	258	6442	70877	<null>	<null>	<null>

È possibile visualizzare e provare le seguenti query in SQL [Cliccando qui](#)
Oppure al seguente link <https://paste.ee/r/A2DEj/0>

Sempre sulla base regionale è possibile visualizzare il numero dei **deceduti**:

Query

```
SELECT sum(DECEDUTI) AS TOTALE_DECEDUTI
FROM REGIONI_COVID19
WHERE DATA ='2020-03-25T17:00:00'
ORDER BY DENOMINAZIONE_REGIONE ASC ;
```

Tabella

	TOTALE_DECEDUTI
1	7503

e degli **Attualmente Positivi**:

Query

```
SELECT sum(TOTALE_POSITIVI) AS TOTALE_ATTUALMENTE_POSITIVI
FROM REGIONI_COVID19
WHERE DATA ='2020-03-25T17:00:00'
ORDER BY DENOMINAZIONE_REGIONE ASC ;
```

Tabella

	TOTALE_ATTUALMENTE_POSITIVI
1	57521

I due dati più rilevanti e purtroppo più preoccupanti dell'andamento dell'epidemia.

La voce «Totale_Casi», indica dunque quante persone, dall'inizio dell'epidemia, hanno di sicuro contratto il virus.

Delle persone che hanno contratto il virus, 7.503 sono morte, mentre 9.362 sono state dimesse «o» guarite (**Attenzione** a questo «o»: è importante).

Di qui si giunge a un dato delicato: quello sul «totale attualmente positivi»: 57.521. È il numero delle persone che il 25 marzo, in Italia, sono positive al Coronavirus, e non sono ancora guarite.

Quel dato viene poi «spacchettato» in tre parti: quanti di quei contagiati si trovano a casa, in isolamento domiciliare; quanti sono in ospedale in terapia intensiva; quanti sono in ospedale, con sintomi ma non in terapia intensiva. In particolare, il dato sulle terapie intensive è cruciale: perché i posti in terapia intensiva non sono infiniti, e perché se si arrivasse a «saturare» il sistema sanitario si correrebbero rischi enormi, per tutti.

Come visto nel precedente paragrafo, con la base di dati in esame è possibile effettuare numerose query che analizzano il fenomeno. Si è potuto ricavare una sorta di classifica delle 5 regioni con più decessi fino ad una determinata data, le 10 province con più casi fino ad una determinata data, fino a poter calcolare anche i casi totale di tutto il mondo.

Il calcolo che più viene commentato nel mondo televisivo ed analizzato nei talk show è l'andamento completo dell'epidemia, contando i totale ospedalizzati, i totale positivi, i totale casi, i totale deceduti, totale dimessi guariti, di seguito è possibile visualizzare il grafico con la ripetizione del codice SQL;

Query

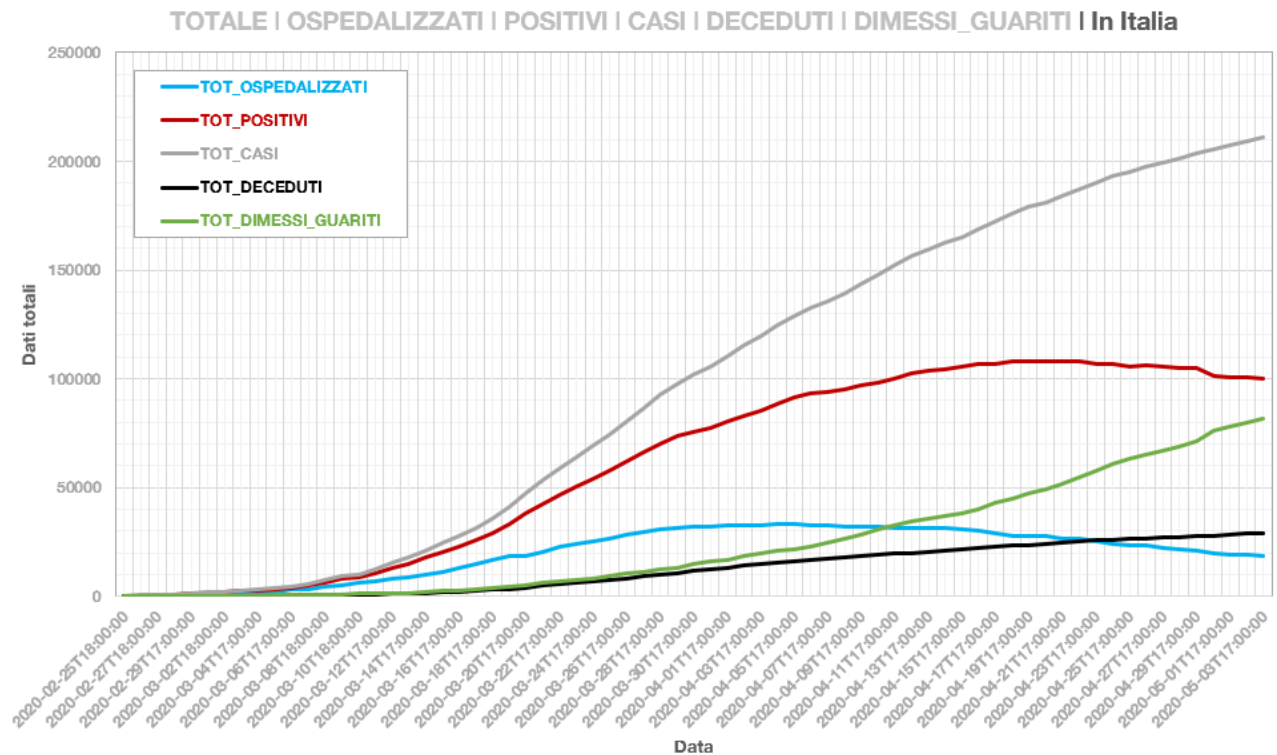
```
SELECT DATA, sum(TOTALE OSPEDALIZZATI) AS TOT_OSP, sum(TOTALE POSITIVI) AS
Tot_pos, sum(TOTALE CASI) AS Tot_cas, sum(DECEDUTI) AS Tot_deceduti,
sum(DIMESSI GUARITI) AS Tot_dim
FROM REGIONI_COVID19
GROUP BY DATA
ORDER BY DATA ASC ;
```

Tabella

	DATA	TOT_OSP	TOT_POS	TOT_CAS	TOT_DECEDUTI	TOT_DIM
1	2020-02-25T18:00:00	149	311	322	10	1
2	2020-02-26T18:00:00	164	385	400	12	3
3	2020-02-27T18:00:00	304	588	650	17	45
4	2020-02-28T18:00:00	409	821	888	21	46
5	2020-02-29T17:00:00	506	1049	1128	29	50
6	2020-03-01T17:00:00	779	1577	1694	34	83
7	2020-03-02T18:00:00	908	1835	2036	52	149
8	2020-03-03T18:00:00	1263	2263	2502	79	160
9	2020-03-04T17:00:00	1641	2706	3089	107	276
10	2020-03-05T17:00:00	2141	3296	3858	148	414
11	2020-03-06T17:00:00	2856	3916	4636	197	523
12	2020-03-07T18:00:00	3218	5061	5883	233	589
13	2020-03-08T18:00:00	4207	6387	7375	366	622
14	2020-03-09T18:00:00	5049	7985	9172	463	724
15	2020-03-10T18:00:00	5915	8514	10149	631	1004
16	2020-03-11T17:00:00	6866	10590	12462	827	1045
17	2020-03-12T17:00:00	7803	12839	15113	1016	1258
18	2020-03-13T17:00:00	8754	14955	17660	1266	1439
19	2020-03-14T17:00:00	9890	17750	21157	1441	1966
20	2020-03-15T17:00:00	11335	20603	24747	1809	2335
21	2020-03-16T17:00:00	12876	23073	27980	2158	2749
22	2020-03-17T17:00:00	14954	26062	31506	2503	2941
23	2020-03-18T17:00:00	16620	28710	35713	2978	4025
24	2020-03-19T17:00:00	18255	33190	41035	3405	4440
25	2020-03-20T17:00:00	18675	37860	47021	4032	5129
26	2020-03-21T17:00:00	20565	42681	53578	4825	6072
27	2020-03-22T17:00:00	22855	46638	59138	5476	7024
28	2020-03-23T17:00:00	23896	50418	63927	6077	7432
29	2020-03-24T17:00:00	25333	54030	69176	6820	8326
30	2020-03-25T17:00:00	26601	57521	74386	7503	9362

La tabella continua fino al 3 Maggio

Grafico a pag.49



Il grafico rappresentato è proprio quello che in televisione, sul web e sui social vediamo. È questo è proprio uno degli indici di decisione ricavati dalla base di dati in esame.

Capitolo 6

Livello dati

Tale livello deve fornire i servizi per la gestione efficiente dei dati. Nel caso in esame, esso sarà costituito dal DBMS ORACLE e dalla base di dati progettata, nonché da una serie di applicazioni memorizzate (***stored procedure e trigger***) nel sistema di basi di dati, al fine di rendere l'informazione quanto più possibile indipendente dalle applicazioni di livello superiore.

In tale capitolo si è fatta anche qui una guida sulla lettura e l'uso dei dati della base di dati.

6.2 Guida alla lettura dei dati attraverso procedure PL/SQL

Un punto molto importante, ora, riguarda la risposta alla domanda: quante persone sono state contagiate, il 25 marzo, in Italia? O, meglio: quante persone sono state trovate positive al coronavirus, il 25 marzo, in Italia?

Ipotizziamo che, il 25 marzo, il numero di persone attualmente positive sia esattamente identico a quello del 24 marzo, 57.521, così come quello dei deceduti (7.503) ma sia cresciuto il numero di dimessi (di 100 unità: da 9.362 a 9.462). La situazione sarebbe la seguente:

Giorno 1: Totale attualmente positivi 57.521; dimessi/guariti 9.362; deceduti 7.503, per un totale di casi di 74.386.

Giorno 2: Totale attualmente positivi 57.521; dimessi/guariti 9.462; deceduti 7.503, per un totale di casi di 74.486.

Quante persone si sono ammalate, tra il 24 marzo e il 25 marzo? La risposta è 100: 100 persone sono uscite dall'ospedale o da casa, e sono guarite, e altrettante sono entrate in ospedale o sono state messe in isolamento domiciliare. Ecco perché il numero di persone attualmente positive è identico a ieri, e quello di guariti è aumentato.

Per trovare quel dato non possiamo fare la differenza tra i «totali attualmente positivi» (darebbe zero, nel caso posto sopra: $57.521 - 57.521$: ma potrebbe anche finire in negativo, ma dobbiamo farla tra i «casi totali» (nel caso sopra: $74.486 - 74.386 = 100$ persone contagiate).

La Protezione civile fornisce però, in diretta, la differenza tra i «totale attualmente positivi». Perché? Perché è un dato importante. Non risponde alla domanda «Quante persone si sono ammalate ieri?», ma alla domanda: il numero di persone positive, in Italia, è aumentato o no? Questo dato dà dunque una indicazione importante sullo stato di salute del nostro sistema sanitario.

Dunque, *quante persone sono state trovate positive al coronavirus, nelle ultime ore, in Italia?*
 R: **Casi totali di oggi - casi totali di ieri**, di seguito il codice **PL/SQL** relativo:

Codice PL/SQL

```
create or replace procedure diff_tot_att_pos IS
--definisco che il cursore che conterrà i casi totali nella data X
cursor totpos is
    SELECT DATA, sum(TOTALE_CASI) AS Totale_Casi
    FROM REGIONI_COVID19
    GROUP BY DATA
    HAVING DATA <= '2020-03-25T17:00:00' AND DATA >= '2020-03-24T17:00:00' ;

--definisco tutte le variabili che serviranno per i calcoli
rec_casi totpos%ROWTYPE; --questa variabile è dello stesso tipo del cursore
giorno1 number;
giorno2 number;
diff number;

BEGIN
    OPEN totpos; --apro il cursore
    FETCH totpos into rec_casi; --inserisco la tupla seguente nella variabile che poi userò per i
    calcoli dopo
    giorno1 := rec_casi.Totale_Casi;
    FETCH totpos into rec_casi;
    giorno2 := rec_casi.Totale_Casi;
    diff := giorno1 - giorno2;
    DBMS_OUTPUT.PUT_LINE('Numero di persone colpite dal COVID_19 = ' || diff);
    CLOSE totpos; --chiudo il cursore
end;
```

Risultato

```
call diff_tot_att_pos();
DB_COVID> call diff_tot_att_pos()

[2020-06-07 10:24:36] completed in 0 ms
[2020-06-07 10:24:36] Numero di persone colpite dal COVID_19 = 5210
```

Importantissimo - lo ripetiamo - è il conteggio sulle terapie intensive: più quel dato si alza, di giorno in giorno, e peggio è.

Ad esempio, *quante persone sono **entrate in terapia intensiva** il giorno 25 marzo?* di seguito il codice **PL/SQL** relativo:

Codice PL/SQL

```
create or replace procedure diff_tot_ter_intensiva IS
--definisco il cursore che conterrà il numero dei ricoveri in terapia intensiva nella data X
cursor totint is
  SELECT DATA, sum(TERAPIA_INTENSIVA) AS Totale_intensiva
  FROM REGIONI_COVID19
  GROUP BY DATA
  HAVING DATA <= '2020-03-25T17:00:00' AND DATA >= '2020-03-24T17:00:00' ;

--dichiaro le variabili che mi serviranno per i calcoli
rec_casi totint%ROWTYPE; --questa variabile è dello stesso tipo del cursore
giorno1 number;
giorno2 number;
diff number;

BEGIN
  OPEN totint; --apro il cursore
  FETCH totint into rec_casi; --inserisco la tupla seguente nella variabile che poi userò per i
calcoli dopo
  giorno1 := rec_casi.Totale_intensiva;
  FETCH totint into rec_casi;
  giorno2 := rec_casi.Totale_intensiva;
  diff := giorno1 - giorno2;
  DBMS_OUTPUT.PUT_LINE('Variazione di persone che sono entrate in terapia intensiva a
causa del COVID_19 = ' || diff);
  CLOSE totint; --chiudo il cursore
end;
```

Risultato

```
call diff_tot_ter_intensiva();
DB_COVID> call diff_tot_ter_intensiva()

[2020-06-07 10:41:21] completed in 0 ms
[2020-06-07 10:41:21] Variazione di persone che sono entrate in terapia intensiva dal
COVID_19 = 93
```


Altro Importantissimo dato è il conteggio sui dimessi guariti, in questo caso, più quel dato si alza, di giorno in giorno, meglio è.

Quante persone sono **guarite** dal coronavirus in Italia nel giorno X? di seguito il codice **PL/SQL** relativo:

Codice PL/SQL

```
create or replace procedure diff_tot_guariti IS
--definisco il cursore che conterrà i guariti nella data X
cursor totgua is
  SELECT DATA, sum(DIMESSI_GUARITI) AS Totale_dimessi
  FROM REGIONI_COVID19
  GROUP BY DATA
  HAVING DATA <= '2020-03-25T17:00:00' AND DATA >= '2020-03-24T17:00:00' ;

--dichiaro le variabili che mi serviranno per i calcoli
rec_casi totgua%ROWTYPE; --questa variabile è dello stesso tipo del cursore
giorno1 number;
giorno2 number;
diff number;

BEGIN
  OPEN totgua; --apro il cursore
  FETCH totgua into rec_casi; --inserisco la tupla seguente nella variabile che poi userò per i
calcoli dopo
  giorno1 := rec_casi.Totale_dimessi;
  FETCH totgua into rec_casi;
  giorno2 := rec_casi.Totale_dimessi;
  diff := giorno1 - giorno2;
  DBMS_OUTPUT.PUT_LINE('Numero di persone GUARITE dal COVID_19 = ' || diff);
  CLOSE totgua; --chiudo il cursore
end;
```

Risultato

```
call diff_tot_guariti();
DB_COVID> call diff_tot_guariti()

[2020-06-07 10:42:08] completed in 0 ms
[2020-06-07 10:42:08] Numero di persone GUARITE dal COVID_19 = 1036
```

Dalla base di dati è possibile fare un'analisi più dettagliata del fenomeno effettuando calcoli determinanti che descrivono il comportamento dell'epidemia.

6.2.1 Calcolo percentuale positivi/tamponi in Italia

Codice PL/SQL

```
create or replace procedure diff_tasso_pos_tam IS
--definisco il cursore che conterrà tutti i positivi e tutti i tamponi effettuati alla data X
cursor totpostam is
    SELECT DATA, sum(TOTALE_POSITIVI) AS TOTALE_POSITIVI_ITA, sum(TAMPONI) AS
TOTALE_TAMPONI
    FROM REGIONI_COVID19
    GROUP BY DATA
    HAVING DATA <= '2020-03-25T17:00:00' AND DATA >= '2020-03-12T17:00:00'
    ORDER BY DATA DESC;

--dichiaro le variabili che mi serviranno per i calcoli
rec_casi totpostam%ROWTYPE; --questa variabile è dello stesso tipo del cursore
tot_pos number;
tot_tamp number;
diff number;

BEGIN
    OPEN totpostam; --apro il cursore
    DBMS_OUTPUT.PUT_LINE('Data : ' || ' | Tasso positivi/popolazione in ITALIA : ');
    LOOP
        FETCH totpostam into rec_casi; --inserisco la tupla seguente nella variabile che userò per
i calcoli dopo
        EXIT when totpostam%NOTFOUND; --se non trova una tupla non stampa nulla ed esce
dal ciclo
        tot_pos := rec_casi.TOTALE_POSITIVI_ITA;
        tot_tamp := rec_casi.TOTALE_TAMPONI;
        diff := (tot_pos / tot_tamp) * 100;
        DBMS_OUTPUT.PUT_LINE(rec_casi.DATA || ' | ' || diff || '%');
    END LOOP;
    CLOSE totpostam; --chiudo il cursore

end;
```

Risultato

```
call diff_tasso_pos_tam();

DB_COVID> call diff_tasso_pos_tam()
[2020-06-07 10:43:04] completed in 16 ms
[2020-06-07 10:43:04] Data : 2020-03-25T17:00:00 Tasso positivi/tamponi in ITALIA : 17,729%
[2020-06-07 10:43:04] Data : 2020-03-24T17:00:00 Tasso positivi/tamponi in ITALIA : 18,194%
[2020-06-07 10:43:04] Data : 2020-03-23T17:00:00 Tasso positivi/tamponi in ITALIA : 18,302%
[2020-06-07 10:43:04] Data : 2020-03-22T17:00:00 Tasso positivi/tamponi in ITALIA : 18,048%
[2020-06-07 10:43:04] Data : 2020-03-21T17:00:00 Tasso positivi/tamponi in ITALIA : 18,300%
[2020-06-07 10:43:04] Data : 2020-03-20T17:00:00 Tasso positivi/tamponi in ITALIA : 18,299%
[2020-06-07 10:43:04] Data : 2020-03-19T17:00:00 Tasso positivi/tamponi in ITALIA : 18,158%
[2020-06-07 10:43:04] Data : 2020-03-18T17:00:00 Tasso positivi/tamponi in ITALIA : 17,343%
[2020-06-07 10:43:04] Data : 2020-03-17T17:00:00 Tasso positivi/tamponi in ITALIA : 17,531%
[2020-06-07 10:43:04] Data : 2020-03-16T17:00:00 Tasso positivi/tamponi in ITALIA : 16,724%
```

```
[2020-06-07 10:43:04] Data : 2020-03-15T17:00:00 Tasso positivi/tamponi in ITALIA : 16,495%
[2020-06-07 10:43:04] Data : 2020-03-14T17:00:00 Tasso positivi/tamponi in ITALIA : 16,259%
[2020-06-07 10:43:04] Data : 2020-03-13T17:00:00 Tasso positivi/tamponi in ITALIA : 15,340%
[2020-06-07 10:43:04] Data : 2020-03-12T17:00:00 Tasso positivi/tamponi in ITALIA : 14,927%
```

6.2.2 Calcolo percentuale positivi/popolazione in Italia

Codice PL/SQL

```
create or replace procedure diff_tasso_pos_pop IS
--definisco il cursore che conterrà la popolazione italiana e tutti i casi di COVID alla data X
cursor totpospop is
    SELECT DATA, SUM(NUM_RESIDENTI) AS POPOLOITA, SUM(TOTALE_CASI) AS
TOT_POS
    FROM REGIONI_COVID19 R JOIN REGIONI R2 on R.DENOMINAZIONE_REGIONE =
R2.DENOMINAZIONE_REGIONE
    GROUP BY DATA
    HAVING DATA <='2020-03-25T17:00:00' AND DATA >= '2020-02-25T18:00:00'
    ORDER BY DATA DESC ;

--dichiaro le variabili che mi serviranno per i calcoli
rec_casi totpospop%ROWTYPE; --questa variabile è dello stesso tipo del cursore
tot_pos number;
pop_ita number;
diff number;

BEGIN
    OPEN totpospop; --apro il cursore
    DBMS_OUTPUT.PUT_LINE('Data : ' || '          | Tasso positivi/popolazione in ITALIA : ');
    LOOP
        FETCH totpospop into rec_casi; --Inserisco la tupla seguente nella variabile che userò per
i calcoli dopo
        EXIT when totpospop%NOTFOUND; --se non trova una tupla non stampa nulla ed esce
dal ciclo
        tot_pos := rec_casi.TOT_POS;
        pop_ita := rec_casi.POPOLOITA;
        diff := (tot_pos / pop_ita) * 100;
        DBMS_OUTPUT.PUT_LINE(rec_casi.DATA || ' | ' || diff || '%');
    END LOOP;
    CLOSE totpospop; --chiudo il cursore

end;
```

Risultato

```
call diff_tasso_pos_pop();
DB_COVID> call diff_tasso_pos_pop()

[2020-06-07 10:46:45] completed in 16 ms
[2020-06-07 10:46:45] Data :          | Tasso positivi/popolazione in ITALIA :
[2020-06-07 10:46:45] 2020-03-25T17:00:00 | 0,020%
[2020-06-07 10:46:45] 2020-03-24T17:00:00 | 0,018%
[2020-06-07 10:46:45] 2020-03-23T17:00:00 | 0,017%
```

```
[2020-06-07 10:46:45] 2020-03-22T17:00:00 | 0,015%
[2020-06-07 10:46:45] 2020-03-21T17:00:00 | 0,014%
[2020-06-07 10:46:45] 2020-03-20T17:00:00 | 0,012%
[2020-06-07 10:46:45] 2020-03-19T17:00:00 | 0,011%
[2020-06-07 10:46:45] 2020-03-18T17:00:00 | 0,007%
[2020-06-07 10:46:45] 2020-03-17T17:00:00 | 0,007%
[2020-06-07 10:46:45] 2020-03-16T17:00:00 | 0,006%
[2020-06-07 10:46:45] 2020-03-15T17:00:00 | 0,005%
[2020-06-07 10:46:45] 2020-03-14T17:00:00 | 0,004%
[2020-06-07 10:46:45] 2020-03-13T17:00:00 | 0,003%
[2020-06-07 10:46:45] 2020-03-12T17:00:00 | 0,003%
[2020-06-07 10:46:45] 2020-03-11T17:00:00 | 0,002%
[2020-06-07 10:46:45] 2020-03-10T18:00:00 | 0,002%
[2020-06-07 10:46:45] 2020-03-09T18:00:00 | 0,002%
[2020-06-07 10:46:45] 2020-03-08T18:00:00 | 0,001%
[2020-06-07 10:46:45] 2020-03-07T18:00:00 | 0,001%
[2020-06-07 10:46:45] 2020-03-06T17:00:00 | 0,0009%
[2020-06-07 10:46:45] 2020-03-05T17:00:00 | 0,0007%
[2020-06-07 10:46:45] 2020-03-04T17:00:00 | 0,0005%
[2020-06-07 10:46:45] 2020-03-03T18:00:00 | 0,0005%
[2020-06-07 10:46:45] 2020-03-02T18:00:00 | 0,0002%
[2020-06-07 10:46:45] 2020-03-01T17:00:00 | 0,0002%
[2020-06-07 10:46:45] 2020-02-29T17:00:00 | 0,0002%
[2020-06-07 10:46:45] 2020-02-28T18:00:00 | 0,00006%
[2020-06-07 10:46:45] 2020-02-27T18:00:00 | 0,00005%
[2020-06-07 10:46:45] 2020-02-26T18:00:00 | 0%
[2020-06-07 10:46:45] 2020-02-25T18:00:00 | 0%
```

6.2.3 Calcolo percentuale positivi/popolazione nel mondo

Codice PL/SQL

```
create or replace procedure diff_tasso_pos_pop_mondo IS
--definisco il cursore che conterrà la popolazione degli stati con più positivi
cursor totpospop_m is
    SELECT * FROM ( SELECT DENOMINAZIONI_STATO, sum(CASI) AS TOTALE_CASI,
POPOLAZIONE
        FROM STATI_COVID19 JOIN STATI S on STATI_COVID19.STATO = S.STATO
        GROUP BY DENOMINAZIONI_STATO, POPOLAZIONE
        ORDER BY TOTALE_CASI desc )
    WHERE ROWNUM <= 10;

--dichiaro le variabili che mi serviranno per i calcoli
rec_casi totpospop_m%ROWTYPE; --questa variabile è dello stesso tipo del cursore
tot_pos number;
pop_mon number;
diff number;

BEGIN
    OPEN totpospop_m; --apro il cursore
    DBMS_OUTPUT.PUT_LINE(RPAD('Stato : ',13)|| ' | Tasso positivi/popolazione nel
MONDO : ');
    LOOP
```

```

    FETCH totpospop_m into rec_casi; --inserisco la tupla seguente nella variabile che userò
per i calcoli dopo
    EXIT when totpospop_m%NOTFOUND; --se non trova una tupla non stampa nulla ed
esce dal ciclo
    tot_pos := rec_casi.TOTALE_CASI;
    pop_mon := rec_casi.POPOLAZIONE;
    diff := (tot_pos / pop_mon) * 100;
    DBMS_OUTPUT.PUT_LINE(RPAD(rec_casi.DENOMINAZIONI_STATO,25) || ' | ' || diff ||
'%');
    END LOOP;
    CLOSE totpospop_m; --chiudo il cursore
end;
```

Risultato

```

call diff_tasso_pos_pop_mondo();
DB_COVID> call diff_tasso_pos_pop_mondo()

[2020-06-07 10:54:16] completed in 15 ms
[2020-06-07 10:54:16] Stato = United_States_of_America Tasso positivi/popolazione : 0,355%
[2020-06-07 10:54:16] Stato = Spain Tasso positivi/popolazione : 0,439%
[2020-06-07 10:54:16] Stato = Italy Tasso positivi/popolazione : 0,336%
[2020-06-07 10:54:16] Stato = United_Kingdom Tasso positivi/popolazione : 0,269%
[2020-06-07 10:54:16] Stato = Germany Tasso positivi/popolazione : 0,201%
[2020-06-07 10:54:16] Stato = France Tasso positivi/popolazione : 0,194%
[2020-06-07 10:54:16] Stato = Turkey Tasso positivi/popolazione : 0,153%
[2020-06-07 10:54:16] Stato = Russia Tasso positivi/popolazione : 0,084%
[2020-06-07 10:54:16] Stato = Brazil Tasso positivi/popolazione : 0,046%
[2020-06-07 10:54:16] Stato = Iran Tasso positivi/popolazione : 0,116%
```

In definitiva questi sono i calcoli più discussi in ambito televisivo e non ma ci sono dati molto più importanti che sono stati talvolta oscurati e né parlati.

Questi dati sono relativamente deterministici della vita del COVID19 e del comportamento del popolo di seguito la trattazione.

6.3 Il calcolo dell'R0 dell'epidemia

Un parametro importante in un'epidemia di una malattia infettiva è il cosiddetto R0 ovvero il “numero di riproduzione di base” che rappresenta il numero medio di infezioni secondarie prodotte da ciascun individuo infetto in una popolazione completamente suscettibile cioè mai venuta a contatto con il nuovo patogeno emergente. Questo parametro misura la potenziale trasmissibilità di una malattia infettiva.

In altre parole, se l'R0 di una malattia infettiva è circa 2, significa che in media un singolo malato infetterà due persone. Quanto maggiore è il valore di R0 e tanto più elevato è il rischio di diffusione dell'epidemia. Se invece il valore di R0 fosse inferiore ad 1 ciò significa che l'epidemia può essere contenuta.

Si rimanda al lettore per un'ulteriore analisi del fenomeno attraverso un modello più approfondito dei dati al seguente link: <https://web.stanford.edu/~jhj1/teachingdocs/Jones-on-R0.pdf>

Il parametro R0 è possibile ricavarlo attraverso modelli matematici (e.g SIR) che determinano sempre in maniera statistica e matematica un possibile andamento dell'epidemia.

Tuttavia si è pensato dunque di effettuare questi calcoli con l'uso del PL/SQL e della basi di dati componendo una formula *semplificata* e ovviamente se ne è tratto un risultato con un tasso di errore aleatorio $\approx 0,10$ o $\approx 0,05$ quindi circa del 10% o del 5%.

La formula semplificata è la seguente:

$$R0 = \frac{\text{totale_positivi}(x)}{\text{totale_positivi}(x - n)}$$

x = indica il determinato giorno

x - n = indica quanti giorni di differenza n rispetto ad x

totale_positivi = indica i totali positivi in Italia in quei determinati giorni.

Di seguito il codice **PL/SQL** e il risultato descritto attraverso un **grafico**.

Codice PL/SQL

```

create or replace procedure calcoloR0 IS
--definisco il cursore che conterrà tutti i positivi alla data X
cursor totcont is
  SELECT DATA, sum(TOTALE_POSITIVI) AS TOTALE_POSITIVI_ITA
  FROM REGIONI_COVID19
  GROUP BY DATA
  HAVING DATA <= '2020-05-03T17:00:00' AND DATA >= '2020-02-25T17:00:00'
  ORDER BY DATA DESC ;
--dichiaro le variabili che mi serviranno per i calcoli
rec_casi totcont%ROWTYPE; -- questa variabile è dello stesso tipo del cursore
data_inizio varchar2(19);
data_fine varchar2(19);
totpos number;
totpos2 number;
R0 number;
x number;
n number;
BEGIN
  OPEN totcont; --apro il cursore
  n := 1;
  SELECT count(*) into x FROM USER_TABLES WHERE table_name = 'R0_COVID19';
  if (x = 0) then
    --creo una tabella dove inserirò i risultati del calcolo dell'R0
    execute immediate ' CREATE TABLE R0_COVID19 (
      DATA_INIZIO VARCHAR2(19),
      DATA_FINE VARCHAR2(19),
      R0 NUMBER,
      PRIMARY KEY (DATA_INIZIO,DATA_FINE))';
    commit;
  end if;
  FETCH totcont into rec_casi; --inserisco la tupla seguente nella variabile che userò per i calcoli dopo
  totpos := rec_casi.TOTALE_POSITIVI_ITA;
  data_fine := rec_casi.DATA;
  LOOP
    FETCH totcont into rec_casi; --inserisco la tupla seguente nella variabile che userò per i calcoli dopo
    EXIT WHEN totcont%NOTFOUND; --se non trova una tupla non stampa nulla ed esce dal ciclo
    totpos2 := rec_casi.TOTALE_POSITIVI_ITA;
    if(totcont%ROWCOUNT = 5*n) then
      data_inizio := rec_casi.DATA;
      R0 := totpos/totpos2;
      DBMS_OUTPUT.PUT_LINE('Calcolo R0 = ' || R0 );
      execute immediate 'INSERT INTO R0_COVID19 VALUES (:val1,:val2,:val3)' using
data_inizio,data_fine,R0; --inserisco i valori del calcolo nella tabella prima creata
      commit;
      n := n+1;
      totpos := rec_casi.TOTALE_POSITIVI_ITA;
      data_fine := rec_casi.DATA;
    END if ;
  END LOOP;
  CLOSE totcont; --chiudo il cursore
end;
```

Risultato

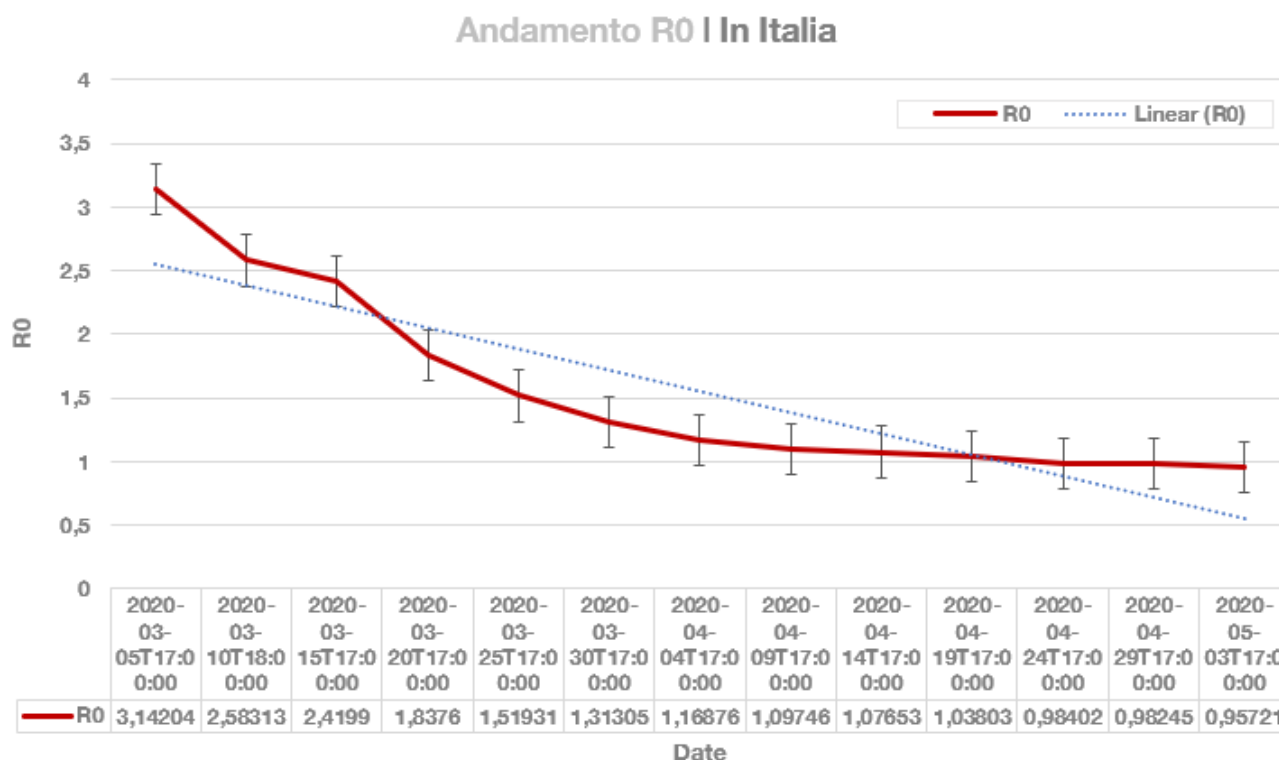
call *calcoloR0()*;

DB_COVID> call *calcoloR0()*

[2020-06-07 11:02:35] completed in 32 ms

	DATA_INIZIO	DATA_FINE	R0
1	2020-04-29T17:00:00	2020-05-03T17:00:00	0.9572126088078198304938991180714142388947
2	2020-04-24T17:00:00	2020-04-29T17:00:00	0.9824457649234466379415547232157105710289
3	2020-04-19T17:00:00	2020-04-24T17:00:00	0.9840195091310492624033549793546837617891
4	2020-04-14T17:00:00	2020-04-19T17:00:00	1.03802820952910605900796808928862509708
5	2020-04-09T17:00:00	2020-04-14T17:00:00	1.07653003292835244691722493471102532077
6	2020-04-04T17:00:00	2020-04-09T17:00:00	1.09745791512789722908217595214899064277
7	2020-03-30T17:00:00	2020-04-04T17:00:00	1.16875860607986442114182819616566041733
8	2020-03-25T17:00:00	2020-03-30T17:00:00	1.31305088576346030145512073851289094418
9	2020-03-20T17:00:00	2020-03-25T17:00:00	1.51930797675647120972002113048071843634
10	2020-03-15T17:00:00	2020-03-20T17:00:00	1.83759646653399990292675823909139445712
11	2020-03-10T18:00:00	2020-03-15T17:00:00	2.41989664082687338501291989664082687339
12	2020-03-05T17:00:00	2020-03-10T18:00:00	2.58313106796116504854368932038834951456
13	2020-02-29T17:00:00	2020-03-05T17:00:00	3.14204003813155386081982840800762631077

Grafico



Come si può notare dal grafico, intorno il 5 e il 10 marzo il virus è incominciato a potenziarsi sempre di più fino a raggiungere un tasso di riproduzione di base del circa 2,9 - 3,3, quindi una persona infettiva poteva infettare 3 persone e così a sua volta.

Da qui dunque nasce l'esigenza delle "regole" (distanza, divieto di assembramenti, uso della mascherina, quarantena, ...) che ormai da quei giorni sono diventati vitali proprio per evitare il contagio e dunque far "scendere" questo determinato parametro.

Infatti, nel passare dei giorni il tasso di riproduzione R0 tenderà verso circa l'1 e seguendo quel determinato andamento questo numero si azzererà se si continuano a rispettare le regole imposte dallo stato.

6.4 Il calcolo del tasso di letalità dell'epidemia

Un altro parametro relativamente importante è il tasso di letalità di epidemia.

*Il **tasso di letalità** indica la proporzione, tipicamente percentuale, di decessi sul totale dei soggetti ammalati in un determinato arco temporale.*

$$L = \frac{N}{P}$$

N = numero totale dei decessi per una determinata malattia in una popolazione per un certo periodo

P = numero di nuovi casi affetti da tale malattia nella stessa popolazione e nello stesso periodo

$$N = \frac{\text{Sum}(\text{Deceduti}) \text{ in Italia}}{7 \text{ (settimana)}}$$

$$P = \frac{\text{Sum}(\text{Nuovi_Positivi}) \text{ in Italia}}{7 \text{ (settimana)}}$$

Di seguito il codice **PL/SQL** e il risultato descritto attraverso un **grafico**.

Codice PL/SQL

```
create or replace procedure calcololetalita IS
--definisco il cursore che conterrà il totale dei deceduti, dei giorni e dei casi alla data X
cursor totcont is
  SELECT DATA, count(DATA) AS SOMMA_GIORNI, SUM(DECEDUTI) AS DECED,
  SUM(TOTALE_CASI) AS TOTCAS
  FROM REGIONI_COVID19 R JOIN REGIONI R2 on R.CODICE_REGIONE =
  R2.CODICE_REGIONE and R.DENOMINAZIONE_REGIONE =
  R2.DENOMINAZIONE_REGIONE
  GROUP BY DATA
  HAVING DATA <='2020-05-03T17:00:00' AND DATA >= '2020-02-25T18:00:00'
  ORDER BY DATA DESC ;

--dichiaro le variabili che mi serviranno per i calcoli
rec_casi totcont%ROWTYPE; --questa variabile è dello stesso tipo del cursore
data_inizio varchar2(19);
data_fine varchar2(19);
L number;
x number;
n number;
num number;
den number;

BEGIN
  OPEN totcont; --apro il cursore
  n := 1;
  SELECT count(*) into x FROM USER_TABLES WHERE table_name =
  'LETALITA_COVID19';
```

```

if (x = 0) then
--creo una tabella dove inserirò i risultati del calcolo del tasso di letalità
execute immediate ' CREATE TABLE LETALITA_COVID19 (
    DATA_INIZIO VARCHAR2(19),
    DATA_FINE VARCHAR2(19),
    L NUMBER,
    PRIMARY KEY (DATA_INIZIO,DATA_FINE));

commit;
end if;
FETCH totcont into rec_casi; --inserisco la tupla seguente nella variabile che userò per i
calcoli dopo
num := rec_casi.DECED / 7;
data_fine := rec_casi.DATA;
LOOP
FETCH totcont into rec_casi; --inserisco la tupla seguente nella variabile che userò per i
calcoli dopo
EXIT WHEN totcont%NOTFOUND; --se non trova una tupla non stampa nulla ed esce dal
ciclo
den := rec_casi.TOTCAS / 7;
if(totcont%ROWCOUNT = 7*n) then
    data_inizio := rec_casi.DATA;
    L := (num/den) * 100;
    DBMS_OUTPUT.PUT_LINE('Calcolo LETALITA = ' || L || '%');
    execute immediate 'INSERT INTO LETALITA_COVID19 VALUES (:val1,:val2,:val3)'
using data_inizio,data_fine,L; --inserisco i valori del calcolo nella tabella prima creata
    commit;
    n := n+1;
    num := rec_casi.DECED / 7;
    data_fine := rec_casi.DATA;
END if ;
END LOOP;
CLOSE totcont; --chiudo il cursore
end;

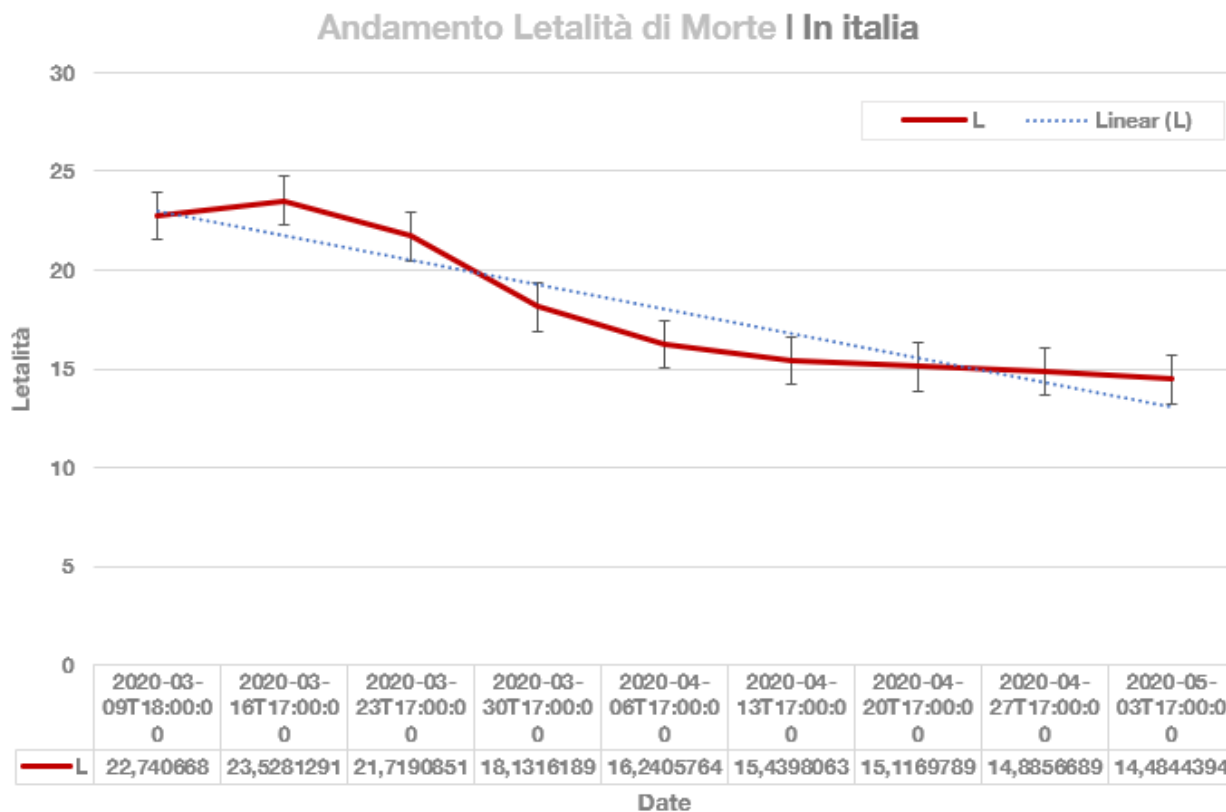
```

Risultato

call *calcololetalita()*;

DB_COVID> call *calcololetalita()*;
 [2020-06-07 11:20:35] completed in 32 ms

	DATA_INIZIO	DATA_FINE	L
1	2020-04-27T17:00:00	2020-05-03T17:00:00	14.48443940746386913657015054108538016...
2	2020-04-20T17:00:00	2020-04-27T17:00:00	14.88566888118833734301542807954620698...
3	2020-04-13T17:00:00	2020-04-20T17:00:00	15.11697886105469043857669450086511697...
4	2020-04-06T17:00:00	2020-04-13T17:00:00	15.439806257403034395346556315872860193
5	2020-03-30T17:00:00	2020-04-06T17:00:00	16.24057637680732069314618779425785588...
6	2020-03-23T17:00:00	2020-03-30T17:00:00	18.13161887778247063056298590579880175...
7	2020-03-16T17:00:00	2020-03-23T17:00:00	21.71908506075768406004288777698355968...
8	2020-03-09T18:00:00	2020-03-16T17:00:00	23.52812908853030963802878325337985172...
9	2020-03-02T18:00:00	2020-03-09T18:00:00	22.74066797642436149312377210216110019...

Grafico


Da come si evince dal grafico, anche qui si nota come tra i giorni del 5 fino al 23 Marzo ci sia stato il picco dell'epidemia con un tasso di mortalità del circa 25%. Nel passare ovviamente dei giorni la letalità incomincia a scendere fino al 15% intorno al 3 Maggio.

Si noti bene che anche questo valore è stato calcolato con una formula semplificata dagli autori della trattazione e dunque anche qui c'è un tasso di errore del circa 3%, 5%.

6.5 Automatizzazione calcolo delle variazioni percentuali rispetto al giorno precedente attraverso l'uso dei trigger

Per automatizzare i calcoli durante l'inserimento dei giorni, si è dunque la necessità di realizzare un **trigger** che calcoli le variazioni percentuali tra un giorno all'altro. Di seguito il codice **PL/SQL**

Codice PL/SQL

```
create or replace trigger var_perc_giornaliere
  before insert on REGIONI_COVID19
  for each row

declare
  regcovid          REGIONI_COVID19%ROWTYPE; --questa variabile ha gli stessi campi della tabella
  REGIONI_COVID19
  var_perc_casi      NUMBER;
  var_perc_terinten  NUMBER;
  var_perc_positivi  NUMBER;
  var_perc_deceduti  NUMBER;
  var_perc_tamponi   NUMBER;

begin
  --prendo i valori registrati nella Regione il giorno precedente
  SELECT * into regcovid FROM ( SELECT *
                                FROM REGIONI_COVID19
                                WHERE DENOMINAZIONE_REGIONE = :NEW.DENOMINAZIONE_REGIONE
                                ORDER BY DATA desc)
  WHERE ROWNUM = 1;

  var_perc_casi := (:NEW.TOTALE_CASI - regcovid.TOTALE_CASI)/100;
  var_perc_terinten := (:NEW.TERAPIA_INTENSIVA - regcovid.TERAPIA_INTENSIVA)/100;
  var_perc_positivi := (:NEW.VARIAZIONE_TOTALE_POSITIVI)/100;
  var_perc_deceduti := (:NEW.DECEDUTI - regcovid.DECEDUTI)/100;
  var_perc_tamponi := (:NEW.TAMPONI - regcovid.TAMPONI)/100;
  DBMS_OUTPUT.PUT_LINE('In ' || :NEW.DENOMINAZIONE_REGIONE);
  DBMS_OUTPUT.PUT_LINE('La variazione percentuale dei casi rispetto ad ieri e" stata del: ' ||
var_perc_casi || '%');
  DBMS_OUTPUT.PUT_LINE('La variazione percentuale dei ricoveri in terapia intensiva rispetto ad ieri
e" stata del: ' || var_perc_terinten || '%');
  DBMS_OUTPUT.PUT_LINE('La variazione percentuale dei positivi rispetto ad ieri e" stata del: ' ||
var_perc_positivi || '%');
  DBMS_OUTPUT.PUT_LINE('La variazione percentuale dei deceduti rispetto ad ieri e" stata del: ' ||
var_perc_deceduti || '%');
  DBMS_OUTPUT.PUT_LINE('La variazione percentuale dei tamponi effettuati rispetto ad ieri e" stata
del: ' || var_perc_tamponi || '%');
end;
```

Attivazione Trigger

```
INSERT INTO
REGIONI_COVID19(data,codice_regione,denominazione_regione,ricoverati_con_sintomi,terapia_intensiv
a,totale_ospedalizzati,isolamento_domiciliare,totale_positivi,variazione_totale_positivi,nuovi_positivi,dimes
si_guariti,deceduti,totale_casi,tamponi,casi_testati,note_it,note_en)
VALUES ('2020-05-04T17:00:00',13,'Abruzzo',301,15,316,1521,1837,-
31,4,831,332,3000,41108,30047,NULL,NULL);
```

DB_COVID> INSERT INTO

*REGIONI_COVID19(data,codice_regione,denominazione_regione,ricoverati_con_sintomi,terapi
a_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 ('2020-05-04T17:00:00',13,'Abruzzo',301,15,316,1521,1837,-
31,4,831,332,3000,41108,30047,NULL,NULL)*

[2020-06-07 13:58:36] 1 row affected in 15 ms

[2020-06-07 13:58:36] In Abruzzo

[2020-06-07 13:58:36] La variazione percentuale dei casi rispetto ad ieri e' stata del: 0,04%

*[2020-06-07 13:58:36] La variazione percentuale dei ricoveri in terapia intensiva rispetto ad ieri
e' stata del: -0,01%*

[2020-06-07 13:58:36] La variazione percentuale dei positivi rispetto ad ieri e' stata del: -0,31%

[2020-06-07 13:58:36] La variazione percentuale dei deceduti rispetto ad ieri e' stata del: 0,02%

*[2020-06-07 13:58:36] La variazione percentuale dei tamponi effettuati rispetto ad ieri e' stata
del: 4,09%*

6.6 Trigger per la sicurezza dei dati

Per garantire la sicurezza dei dati e quindi tutti i vincoli imposti sulla base di dati è necessario realizzare un trigger che mi impedisca che i dati vengano modificati essendo che la base di dati in esame garantisce la non volatilità. Di seguito il codice **PL/SQL**:

Codice PL/SQL

--Impedisce che venga modificata la tabella REGIONI_COVID19

create or replace trigger GEST_MODIFICA_REGIONI

before update

on REGIONI_COVID19

for each row

begin

raise_application_error(-20450,' Non si possono modificare i dati sulle rilevazioni');

end;

----Impedisce che venga modificata la tabella STATI_COVID19-----

create or replace trigger GEST_MODIFICA_STATI

before update

on STATI_COVID19

for each row

begin

raise_application_error(-20451,' Non si possono modificare i dati sulle rilevazioni');

end;

----Impedisce che venga modificata la tabella PROVINCE_COVID19-----

create or replace trigger GEST_MODIFICA_PROVINCE

before update

on PROVINCE_COVID19

for each row

begin

raise_application_error(-20452,' Non si possono modificare i dati sulle rilevazioni');

end;

Attivazione Trigger

update REGIONI_COVID19 set TOTALE_CASI = 232313 where DATA = '2020-02-25T18:00:00' AND DENOMINAZIONE_REGIONE = 'Lombardia';

DB_COVID> update REGIONI_COVID19 set TOTALE_CASI = 232313 where DATA = '2020-02-25T18:00:00' AND DENOMINAZIONE_REGIONE = 'Lombardia'

[2020-06-07 14:10:34] [72000][20450] ORA-20450: Non si possono modificare i dati sulle rilevazioni

È possibile visualizzare e provare le seguenti procedure in PL/SQL [Cliccando qui](#)
Oppure al seguente link <https://paste.ee/r/FTnIU/0>

Capitolo 7

Il livello Applicazione

A livello intermedio, le applicazioni sono scritte usando linguaggi ad alto livello (ad es., Java, C++, ecc.) e sono eseguite come processi separati che si connettono al DBMS per poter interagire con esso. In definitiva, servono a manipolare e gestire il DBMS da un linguaggio ad alto livello, senza che l'utente conosca effettivamente la struttura di fondo della base di dati.

Gli altri oggetti applicativi avranno, invece, il compito di estrarre le entità significative della realtà di interesse nonché di realizzare tutte le operazioni previste sulla base di dati, eventualmente inglobando il codice SQL, e di interagire con i dati al loro interno utilizzando stored procedure (aggiunta di esempi fatti al capitolo dei pl/sql) memorizzate sul DBMS.

A titolo di esempio, si sono progettate ed implementate classi costituenti degli oggetti applicativi per la visualizzazione dei dati di contagio del virus da parte degli utenti attraverso un info desk informativo che consentirà di ottenere tutte le informazioni.

Nell'interazione tra linguaggi di alto livello e sistemi DBMS potrebbero nascere problemi di compatibilità, che spesso producono applicazioni poco reattive o nel peggiore dei casi malfunzionanti. È, quindi, indispensabile nella fase di progettazione di questo livello scegliere tutte le opportune metodologie per consentire un corretto “match” tra le istruzioni dell'uno e dell'altro linguaggio.

Dopo aver discusso delle applicazioni che si interfacciano con il DBMS per trarre le relative informazioni, pare doveroso fare un accenno ai sistemi ERP (Enterprise Resource Planning) che rappresentano per gli organi gestionali uno strumento di primaria importanza. Un ERP rappresenta un programma software capace di integrare tutti moduli applicativi relativi alla produzione e/o erogazione di prodotti e/o servizi in un unico ambiente lavorativo. Ai fini del nostro progetto tale strumento è utile dal momento che l'accesso alla base di dati è fatto da una molteplicità di utenti con diversi ruoli e altrettanti diversi obiettivi, risulterebbe quindi fondamentale avere un ambiente di lavoro integrato composto da diversi moduli ognuno destinato ad uno specifico utente. Per esempio, potremmo avere la seguente suddivisione dei moduli applicativi:

- Organi territoriali (Regioni, Province, etc.);
- Forze dell'ordine;
- Enti governativi (ad esempio, la Protezione Civile);
- Addetti stampa;
- etc.

Ognuno dei suddetti utenti del DataBase avrebbe, quindi, accesso ad un modulo indipendente con tutte le relative operazioni necessarie al suo lavoro, tutti i moduli però lavorano sempre sullo stesso pacchetto di dati che rappresentano il fulcro di tutta l'elaborazione.

Capitolo 8

Il livello di presentazione

In quest'ultimo capitolo verranno trattati gli indici e le viste, due strutture dati di notevole importanza in una base di dati sia per l'organizzazione e l'aggregazione dei dati, sia per migliorare l'efficienza delle operazioni sui dati stessi.

Innanzitutto, bisogna introdurre il concetto di indice e definire tutte le sue varianti, al fine di comprendere i motivi del suo uso nella nostra trattazione, all'interno di un DBMS vengono elaborate tantissime informazioni che per motivi di capacità non possono essere posti tutti in memoria centrale. Allora, un opportuno *Gestore dei Metodi di Accesso e dei File*, si occupa di creare all'interno della memoria centrale pochi file con all'interno i soli record di cui il DBMS ha bisogno in quello specifico istante, tuttavia i record all'interno del file sono spesso in ordine sparso e la loro ricerca è spesso dispendiosa e poco efficiente per i risultati dell'elaborazione.

Ecco che nasce l'esigenza di una struttura dati ordinata che permettesse una rapida ricerca dei record all'interno di un file, quest'ultima è la definizione di **indice**, quello che accade realmente è che per ogni file (ordinato o meno) collocato in memoria principale viene associato un file di indice che contiene: una chiave di ricerca e un identificatore di record.

A seconda del tipo di chiave di ricerca, ovvero del campo su cui viene resa più efficiente la selezione, gli indici possono distinguersi in:

- *indice primario*: l'indice è costruito su una chiave di ricerca che include la chiave primaria di una relazione;
- *indice secondario*: l'indice è costituito su una chiave di ricerca che include una chiave non primaria di una relazione;
- *indice di clustering*: l'indice è costituito su un campo che non è chiave, quindi ad ogni valore presente nell'indice corrispondono più record all'interno del file.

Nell'ottica di un DWH, gli indici rappresentano degli strumenti indispensabili vista la grande mole di dati da gestire, per questo vengono adoperati in maniera rigorosa attraverso due tipi di indici particolari, ovvero:

- *Indici di Bitmap*: usati principalmente quando la chiave di ricerca scelta è un campo con un dominio di valori ridotto, memorizzano quindi un bit per ogni tupla, il bit ha valore 1 se l'attributo ha uno specifico valore, 0 altrimenti;
- *Indici di Join*: ottimizzano le operazioni di join calcolando in anticipo tale operazione e costituendo un indice che poi verrà usato nelle query di analisi.

Nel caso in esame, sono stati usati degli indici secondari per ottimizzare tutte quelle operazioni in cui è richiesto la join su tali attributi, rendendo tutte le query per lo studio del contagio più efficienti.

8.1 Indice sulle relazioni regioni, province, stati

--Indice sulla denominazione della Regione

```
CREATE INDEX idx_info_sett_REGIONI
ON REGIONI (DENOMINAZIONE_REGIONE);
```

--Indice sulla denominazione della Provincia

```
CREATE INDEX idx_info_sett_PROVINCE
ON PROVINCE (DENOMINAZIONE_PROVINCIA);
```



```
--Indice sulla descrizione dello stato
CREATE INDEX idx_info_sett_STATI
ON STATI (DESCRIZIONE);
```

8.2 Le view sul DB_COVID19

A questo punto, non resta che parlare delle viste, queste strutture che permettono di creare una tabella, non realmente esistente nel database, ma virtuali cioè create attraverso una `SELECT` che ogni volta andrà eseguita da capo. Fanno eccezione a quest'ultimo caso tutte le `MATERIALIZED VIEW`, infatti, tali viste si distinguono per il fatto che sono memorizzate nel database come una normale tabella, tuttavia, hanno il grosso svantaggio di essere non aggiornabili ovvero una volta eseguita la select non verranno più visti gli aggiornamenti delle tabelle su cui la vista è definita.

Per questo motivo è stata scelta per la nostra base di dati una vista virtuale (non materializzata), che ogni qual volta verrà invocata rieseguirà la query, il vantaggio dell'uso delle viste risiede nel fatto che rendono alcune query e/o procedure (come vedremo nel seguito), più semplici da scrivere e soprattutto da leggere, creando una sorta di *information hiding* perché all'interno della query, la vista verrà richiamata come una normale tabella del database.

```
--Casi nell'arco di una settimana nelle Regioni
CREATE VIEW info_settimana_REGIONI AS
SELECT *
FROM REGIONI_COVID19
WHERE DATA >='2020-03-25T17:00:00' AND DATA <='2020-03-31T17:00:00'
ORDER BY DENOMINAZIONE_REGIONE ASC;
```

```
--Casi nell'arco di una settimana nelle Province
CREATE VIEW info_settimana_PROVINCE AS
SELECT DATA_REG,DENOMINAZIONE_PROVINCIA,TOTALE_CASI
FROM PROVINCE_COVID19 P JOIN PROVINCE P2 on P.CODICE_PROVINCIA =
P2.CODICE_PROVINCIA
WHERE DATA_REG >='2020-03-25T17:00:00' AND DATA_REG <='2020-03-31T17:00:00'
ORDER BY DENOMINAZIONE_PROVINCIA ASC;
```

```
--Casi nell'arco di una settimana negli stati
CREATE VIEW info_settimana_STATI AS
SELECT DATA_STATO,DENOMINAZIONI_STATO,CASI,DECEDUTI
FROM STATI_COVID19
WHERE DATA_STATO like '%03_2020' and DATA_STATO >='25/03/2020'
ORDER BY DENOMINAZIONI_STATO ASC;
```

8.3 Utilizzo delle view

Procedura che estrae dalla vista **INFO_SETTIMANA_PROVINCE** le informazioni relative alla provincia desiderata

Codice PL/SQL

```
create or replace procedure info_province is
--definisco il cursore che conterrà le informazioni settimanali relative alla provincia che viene fornita in
input
cursor info_covid_cursor (provincia in
INFO_SETTIMANA_PROVINCE.DENOMINAZIONE_PROVINCIA%TYPE)
is SELECT *
FROM INFO_SETTIMANA_PROVINCE
WHERE DENOMINAZIONE_PROVINCIA= provincia
ORDER BY DATA_REG ASC;

--dichiaro la variabile che mi servirà per fornire l'output
info_covid info_covid_cursor%ROWTYPE; --questa variabile è dello stesso tipo del cursore

begin
open info_covid_cursor('Napoli'); --apro il cursore e fornisco la provincia di cui voglio le informazioni
fetch info_covid_cursor into info_covid; --inserisco la tupla seguente nella variabile
DBMS_OUTPUT.PUT_LINE('Ecco le informazioni relative alla provincia di ' ||
info_covid.DENOMINAZIONE_PROVINCIA || ' nella settimana tra il 25 ed il 31 Marzo');
DBMS_OUTPUT.PUT_LINE('Data          | Totale casi ');
loop
exit when info_covid_cursor%notfound; --se non trova una tupla non stampa nulla ed esce dal ciclo
DBMS_OUTPUT.PUT_LINE(RPAD(info_covid.DATA_REG,22) || info_covid.TOTALE_CASI);
fetch info_covid_cursor into info_covid; --inserisco la tupla seguente nella variabile
end loop;
close info_covid_cursor; --chiudo il cursore
end;
```

Risultato

call info_province(); --chiamo la procedura che mi darà le info in output

```
DB_COVID> call info_province()
[2020-06-07 16:02:05] completed in 15 ms
[2020-06-07 16:02:05] Ecco le informazioni relative alla provincia di Napoli nella settimana tra il
25 ed il 31 Marzo
[2020-06-07 16:02:05] Data                               | Totale casi
[2020-06-07 16:02:05] 2020-03-25T17:00:00                626
[2020-06-07 16:02:05] 2020-03-26T17:00:00                665
[2020-06-07 16:02:05] 2020-03-27T17:00:00                734
[2020-06-07 16:02:05] 2020-03-28T17:00:00                827
[2020-06-07 16:02:05] 2020-03-29T17:00:00                898
[2020-06-07 16:02:05] 2020-03-30T17:00:00                991
[2020-06-07 16:02:05] 2020-03-31T17:00:00               1053
```

Procedura che estrae dalla vista **INFO_SETTIMANA_REGIONI** le informazioni relative alla regione desiderata

Codice PL/SQL

```
create or replace procedure info_regioni is
--definisco il cursore che conterrà le informazioni settimanali relative alla regione che viene fornita in input
cursor info_covid_cursor (regione in
INFO_SETTIMANA_REGIONI.DENOMINAZIONE_REGIONE%TYPE)
is SELECT *
FROM INFO_SETTIMANA_REGIONI
WHERE DENOMINAZIONE_REGIONE= regione
ORDER BY DATA ASC;

--dichiaro la variabile che mi servirà per fornire l'output
info_covid info_covid_cursor%ROWTYPE; --questa variabile è dello stesso tipo del cursore

begin
open info_covid_cursor('Liguria'); --apro il cursore e fornisco la regione di cui voglio le informazioni
fetch info_covid_cursor into info_covid; --inserisco la tupla seguente nella variabile
DBMS_OUTPUT.PUT_LINE('Ecco le informazioni relative alla regione ' ||
info_covid.DENOMINAZIONE_REGIONE || ' nella settimana tra il 25 ed il 31 Marzo');
DBMS_OUTPUT.PUT_LINE('Data          | Deceduti | Totale casi | Terapia intensiva ' ||
'| Dimessi guariti | Nuovi positivi | Tamponi');
loop
exit when info_covid_cursor%notfound; --se non trova una tupla non stampa nulla ed esce dal ciclo
DBMS_OUTPUT.PUT_LINE(RPAD(info_covid.DATA,22) || RPAD(info_covid.DECEDUTI,11) ||
RPAD(info_covid.TOTALE_CASI,14) || RPAD(info_covid.TERAPIA_INTENSIVA,20) ||
RPAD(info_covid.DIMESSI_GUARITI,18) || RPAD(info_covid.NUOVI_POSITIVI,17) ||
info_covid.TAMPONI);
fetch info_covid_cursor into info_covid; --inserisco la tupla seguente nella variabile
end loop;
close info_covid_cursor; --chiudo il cursore
end;
```

Risultato

call info_regioni(); --chiamo la procedura che mi darà le info in output

DB_COVID> call info_regioni()

[2020-06-07 16:06:23] completed in 0 ms

[2020-06-07 16:06:23] Ecco le informazioni relative alla regione Liguria nella settimana tra il 25 ed il 31 Marzo

Data	Deceduti	Totale casi	Terapia intensiva	Dimessi guariti	Nuovi positivi	Tamponi
2020-03-25	254	2305	147	225	189	6602
2020-03-26	280	2567	154	260	262	7304
2020-03-27	331	2696	157	305	129	7804
2020-03-28	358	2822	167	378	126	8177
2020-03-29	377	3076	166	420	254	9100
2020-03-30	397	3217	175	437	141	9677
2020-03-31	428	3416	179	480	199	10376

Procedura che estrae dalla vista **INFO_SETTIMANA_STATI** le informazioni relative allo stato che viene fornito in input

Codice PL/SQL

```
create or replace procedure info_stati is
--definisco il cursore che conterrà le informazioni settimanali relative allo stato che viene fornito in input
cursor info_covid_cursor (stato in INFO_SETTIMANA_STATI.DENOMINAZIONI_STATO%TYPE)
is SELECT *
FROM INFO_SETTIMANA_STATI
WHERE DENOMINAZIONI_STATO= stato
order by DATA_STATO;

--dichiaro la variabile che mi servirà per fornire l'output
info_covid info_covid_cursor%ROWTYPE; --questa variabile è dello stesso tipo del cursore

begin
open info_covid_cursor('Germany'); --apro il cursore e fornisco lo stato di cui voglio le informazioni
fetch info_covid_cursor into info_covid; --inserisco la tupla seguente nella variabile
DBMS_OUTPUT.PUT_LINE('Ecco le informazioni relative allo stato ' ||
info_covid.DENOMINAZIONI_STATO);
DBMS_OUTPUT.PUT_LINE('Data      | Totale casi | Deceduti');
loop
exit when info_covid_cursor%notfound; --se non trova una tupla non stampa nulla ed esce dal ciclo
DBMS_OUTPUT.PUT_LINE(RPAD(info_covid.DATA_STATO,13) || RPAD(info_covid.CASI,14) ||
info_covid.DECEDUTI);
fetch info_covid_cursor into info_covid; --inserisco la tupla seguente nella variabile
end loop;
close info_covid_cursor; --chiudo il cursore
end;
```

Risultato

```
call info_stati(); --chiamo la procedura che mi darà le info in output

DB_COVID> call info_stati()
[2020-06-07 16:13:37] completed in 16 ms
[2020-06-07 16:13:38] Ecco le informazioni relative allo stato Germany
[2020-06-07 16:13:38] Data      | Totale casi | Deceduti
[2020-06-07 16:13:38] 25/03/2020 2342      23
[2020-06-07 16:13:38] 26/03/2020 4954      49
[2020-06-07 16:13:38] 27/03/2020 5780      55
[2020-06-07 16:13:38] 28/03/2020 6294      72
[2020-06-07 16:13:38] 29/03/2020 3965      64
[2020-06-07 16:13:38] 30/03/2020 4751      66
[2020-06-07 16:13:38] 31/03/2020 4615     128
```

È possibile visualizzare e provare le seguenti view in SQL e PL/SQL [Cliccando qui](#)
Oppure al seguente link <https://paste.ee/r/39KmH/0>

8.4 Una possibile interfaccia web

Un passo fondamentale della realizzazione della base di dati è proprio l'interfacciamento tra la base stessa e l'utente.

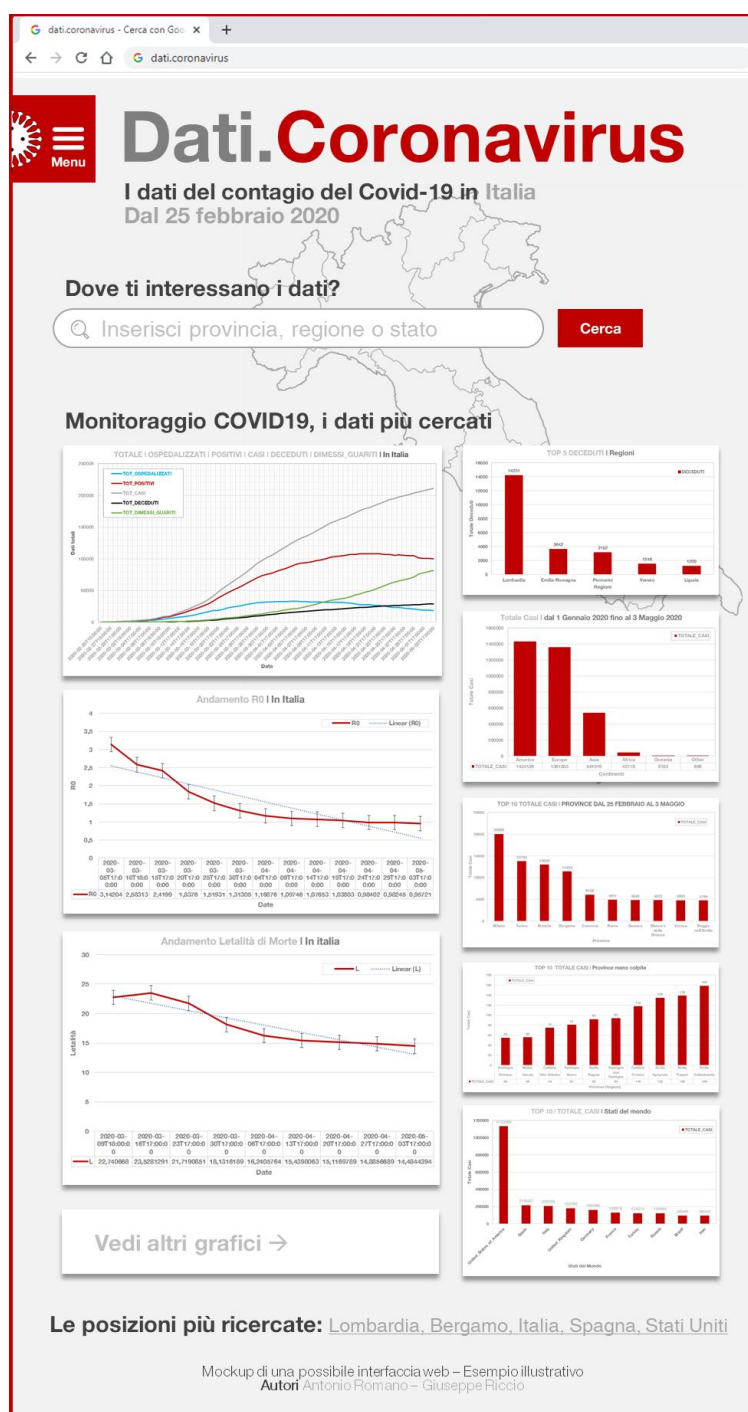
Oggi giorno, la base di dati è diventata “big” grazie all'avvento di Internet e dunque dello sviluppo dello scambio di dati e dallo stragrande utilizzo dei dati che siano multimediali, che siano semplicemente testo o grafici affatto sì di semplificare l'uso e la conoscenza dei dati tramite “interfacce web”, ovvero, un'interazione tra una semplice pagina web (e quindi utilizzo di linguaggi orientati al web design e.g HTML, PHP, JSP ...) e la base di dati installato su un server (o più server: sistemi distribuiti). Dunque nascondere l'utente il lavoro sporco che c'è dietro ad una base di dati. Si è dunque pensato di creare una sorta di mockup di una possibile interfaccia web della nostra base di dati.

Top
Titolo e Menu

Ricerca
Consultare i dati più approfonditamente cercando la provincia, la regione o lo stato

Monitoraggio
Consultare i dati più cercati tramite grafici

Tag
Le posizioni più ricercate



Appendice

Simulatore della base di dati DB_COVID19

Si è realizzato una piccola base di dati su Oracle Live, caratterizzata da tutte le regole imposte su questa trattazione ma con indice di data ristretto per questioni di dimensione (dal 19 Marzo al 25 Marzo) seguendo le linee guida descritte nel **Cap.5**, **Cap.6** e **7**

Grazie ad **Oracle Live Tutorial** è possibile simulare il **run** delle DDL e delle DML definite su questa trattazione.

Effettuando il login ad Oracle Live è possibile utilizzare il simulatore **DB_COVID19**

DB_Covid_19_Tutorial

[View All Tutorials](#)
[Login and Run Tutorial](#)

Tutorial	DB_Covid_19_Tutorial
Description	Si realizza una simulazione che fornisce un opportuno strumento di gestione del territorio e supporto per la gestione dell' emergenza Coronavirus (COVID_19).
Tags	Base di dati, Covid_19, Coronavirus, Simulatore SQL, PL/SQL, Guida alla lettura della base di dati COVID_19
Area	SQL General
Visibility	Unlisted - anyone with the share link can access
Contributor	Antonio Romano, Giuseppe Riccio, Università Federico II di Napoli, Facoltà Ingegneria Informatica.
Created	Friday June 05, 2020
Modules	18

Module 1

Prefazione

Grazie a questo tool di Oracle live si è potuto realizzare la strutturazione della base di dati è orientata nell'area dell'epidemia.

Attenzione, per inserire gli elementi della base di dati cliccare sul triangolino ►

Module 2

La tabella Master

Il dataset fornito da <https://github.com/pcm-dpc/COVID-19> è una rappresentazione della base di dati direzionale (Data WareHouse DWH), si incomincia dunque caricando la tabella **master** dal 19 Marzo al 25 Marzo

```

create table MASTER
(
  DATA_REG          VARCHAR2(19)  not null,
  STATO              VARCHAR2(3)   not null,
  CODICE_REGIONE     NUMBER        not null,
  DENOMINAZIONE_REGIONE VARCHAR2(21) not null,
  CODICE_PROVINCIA   NUMBER        not null,
  ...

```

Si accede nel simulatore dell'Oracle Live cliccando [qui](https://livesql.oracle.com/apex/livesql/file/tutorial_J6ZPO3GWTIOPQBJ48RHTTFJXB.html) o al seguente link https://livesql.oracle.com/apex/livesql/file/tutorial_J6ZPO3GWTIOPQBJ48RHTTFJXB.html

Cliccando sul link e si fa la login dove indicato si avrà la **SQL Worksheet** di **Oracle Live** dove è possibile simulare la base di dati. Ecco come:

Premendo Run il codice presente nel Worksheet sarà eseguito. Il risultato sarà poi mostrato nel riquadro sottostante

SQL Worksheet

Clear
Find
Actions
Save
Run

```

1 create table REGIONI_COVID19
2 (
3     DATA VARCHAR2(19) not null,
4     CODICE_REGIONE NUMBER not null,
5     DENOMINAZIONE_REGIONE VARCHAR2(21) not null,
6     RICOVERATI_CON_SINTOMI NUMBER,
7     TERAPIA_INTENSIVA NUMBER,
8     TOTALE_OSPEDALIZZATI NUMBER,
9     ISOLAMENTO_DOMICILIARE NUMBER,
10    TOTALE_POSITIVI NUMBER,
11    VARIAZIONE_TOTALE_POSITIVI NUMBER,
12    NUOVI_POSITIVI NUMBER,
13    DIMESSI_GUARITI NUMBER,
14    DECEDUTI NUMBER,
15    TOTALE_CASI NUMBER,
16    TAMPONI NUMBER,
17    CASI_TESTATI NUMBER,
18    NOTE_IT VARCHAR2(10),
19    NOTE_EN VARCHAR2(10),
20 )
21

```

3.1 Insert dati dal 19 Marzo al 25 Marzo 2020

► 19 Marzo 2020

▼ 20 Marzo 2020

```

INSERT INTO REGIONI_COVID19 VALUES ('2020-03-20T17:00:00', 1, 'Piemonte', 2279, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0);
INSERT INTO REGIONI_COVID19 VALUES ('2020-03-20T17:00:00', 2, 'Valle d'Aosta', 47, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0);
INSERT INTO REGIONI_COVID19 VALUES ('2020-03-20T17:00:00', 3, 'Lombardia', 7387, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0);
INSERT INTO REGIONI_COVID19 VALUES ('2020-03-20T17:00:00', 4, 'P.A. Bolzano', 87, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0);
INSERT INTO REGIONI_COVID19 VALUES ('2020-03-20T17:00:00', 4, 'P.A. Trento', 169, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0);
INSERT INTO REGIONI_COVID19 VALUES ('2020-03-20T17:00:00', 5, 'Veneto', 771, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0);

```

► 21 Marzo 2020

▼ 22 Marzo 2020

23 Marzo 2020

24 Marzo 2020

25 Marzo 2020

DATA	CODICE_REGIONE	DENOMINAZIONE_REGIONE	RICOVERATI_CON_SINTOMI
2020-03-19T17:00:00	1	Piemonte	2279
2020-03-19T17:00:00	2	Valle d'Aosta	47
2020-03-19T17:00:00	3	Lombardia	7387
2020-03-19T17:00:00	4	P.A. Bolzano	87
2020-03-19T17:00:00	4	P.A. Trento	169
2020-03-19T17:00:00	5	Veneto	771

Regioni_Covid19 si possono analizzare tutti i dati per gli eventuali calcoli dell'andamento del virus.

Proiezione tabella **Regioni_Covid19**

```
select * from REGIONI_COVID19;
```

2020-03-19T17:00:00			
2020-03-19T17:00:00	10	Umbria	69

il risultato dopo il run andato a buon fine!

Sitografia e Bibliografia

Sitografia

- Donna Glamour, Lorenzo Martinotti, 21/02/2020: <https://www.donnaglamour.it/coronavirus-2019-ncov/benessere/>
- Agenzia Italiana, 12/03/2020: <https://www.agi.it/fact-checking/news/2020-03-12/coronavirus-bilancio-morti-contagi-guariti-dati-7447972/>
- Corriere della Sera, Davide Casati, 26/03/2020: https://www.corriere.it/salute/20_marzo_26/come-si-legge-bollettino-protezione-civile-coronavirus-14768dbe-6f32-11ea-b81d-2856ba22fce7.shtml?refresh_ce-cp
- Istituto Superiore di Sanità, 05/02/2020: https://www.iss.it/primo-piano/-/asset_publisher/o4oGR9qmvUz9/content/id/5268851
- Tuttitalia.it, 01/01/2019: <https://www.tuttitalia.it/province/>
- Comuni e città: <https://www.comuniecitta.it/elenco-scuole-per-provincia>
- Ministero della salute, 17/01/2020: <http://www.dati.salute.gov.it/dati/dettaglioDataset.jsp?menu=dati&idPag=2>
- Federalberghi, 03/10/2017: <https://www.federalberghi.it/rapporti/viii-rapporto-sul-sistema-alberghiero-e-turistico-ricettivo-in-italia.aspx#.Xtn2K54zafU>
- Wikipedia, 29/11/2018: https://it.wikipedia.org/wiki/Categoria:Aeroporti_d%27Italia_per_regione
- DatiOpen.it, OpenStreetMap, 10/03/2016: http://www.datiopen.it/it/opendata/Mappa_delle_stazioni_ferrovie_in_Italia
- Anas: <https://www.stradeanas.it/it/le-strade/anas-regione>
- Oracle Italia by Massimo Ruocchio, 20/02/2010: <https://oracleitalia.wordpress.com/tag/rownum/>
- UniTo, Sistemi RAID: <http://www.di.unito.it/~gunetti/DIDATTICA/architettureII/09-raid-2.pdf>
- Calcolo dell'R0 <https://github.com/pcm-dpc/COVID-19/issues/587>
- Letalità https://it.wikipedia.org/wiki/Tasso_di_letalit%C3%A0

Bibliografia

- Sistemi di basi di dati e applicazioni, Angelo Chianese, Vincenzo Moscato, Antonio Picariello, Lucio Sansone, 01/09/2015
- Zausa Francesco, 2010/2011: http://tesi.cab.unipd.it/33135/1/zausa_francesco_454615if_tesi.pdf
- Iannizzi Giovanni, 2015/2016: https://www.unirc.it/documentazione/materiale_didattico/1467_2016_423_24160.pdf