

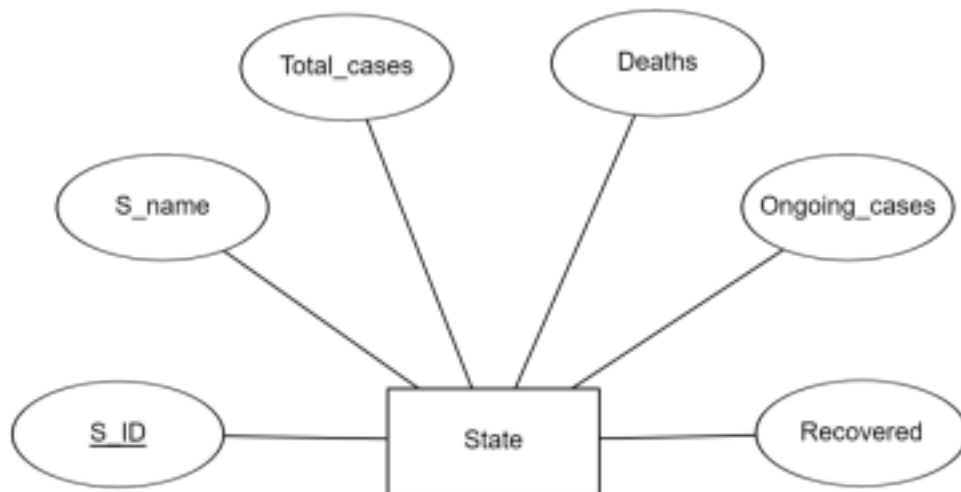
COVID-19 Database Management System

Introduction

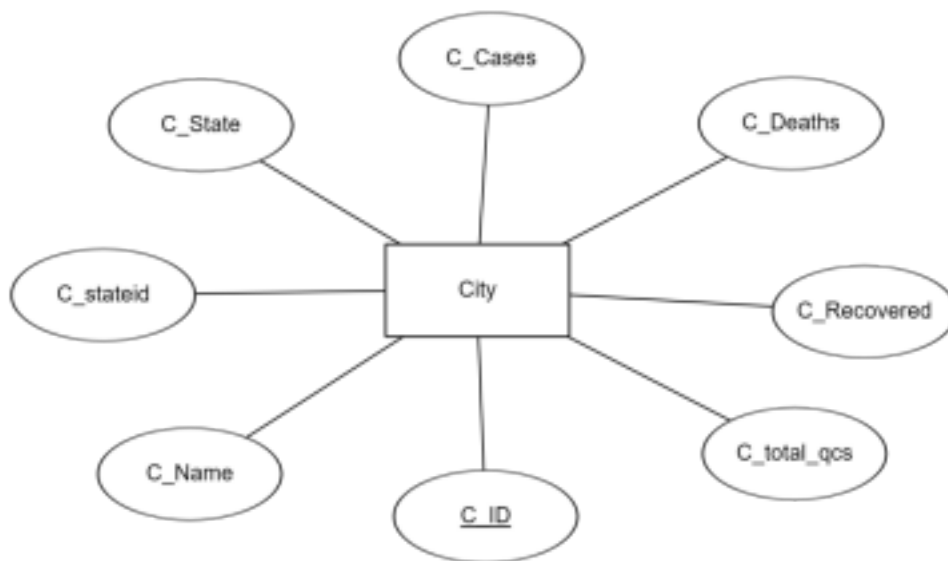
The following database was made for logging purposes to manage and understand data during the covid 19 pandemic on a state level. The purpose of this project is to make information easily accessible for anyone to figure out where their nearest centre is or where their loved one is admitted. The project can be scaled further to add more data and support more states and can also be made to work on an international level

E-R Diagrams

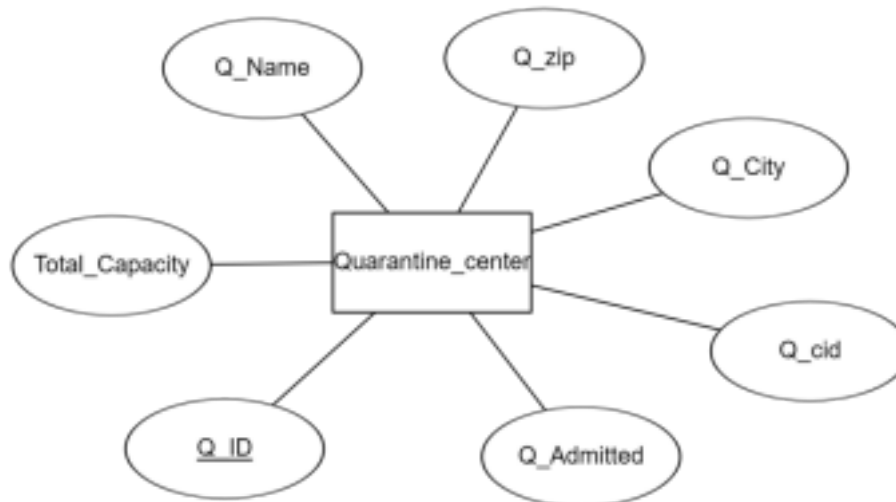
1. STATE



2. CITY

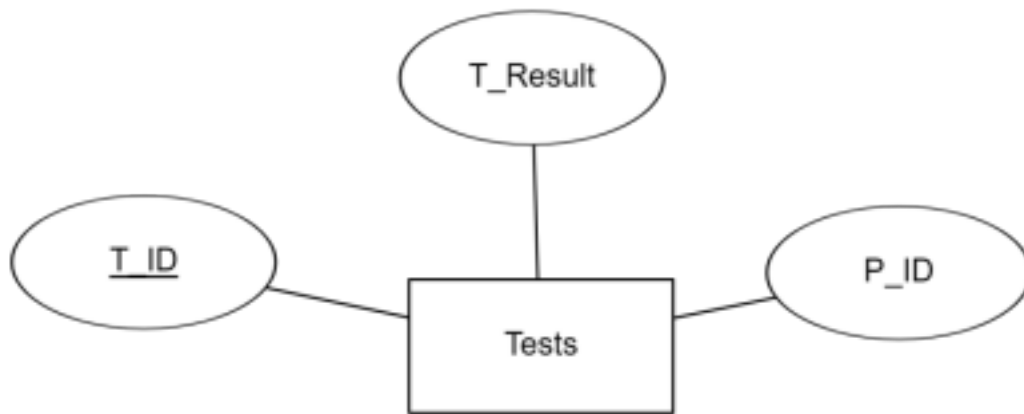


3. QUARANTINE_CENTER

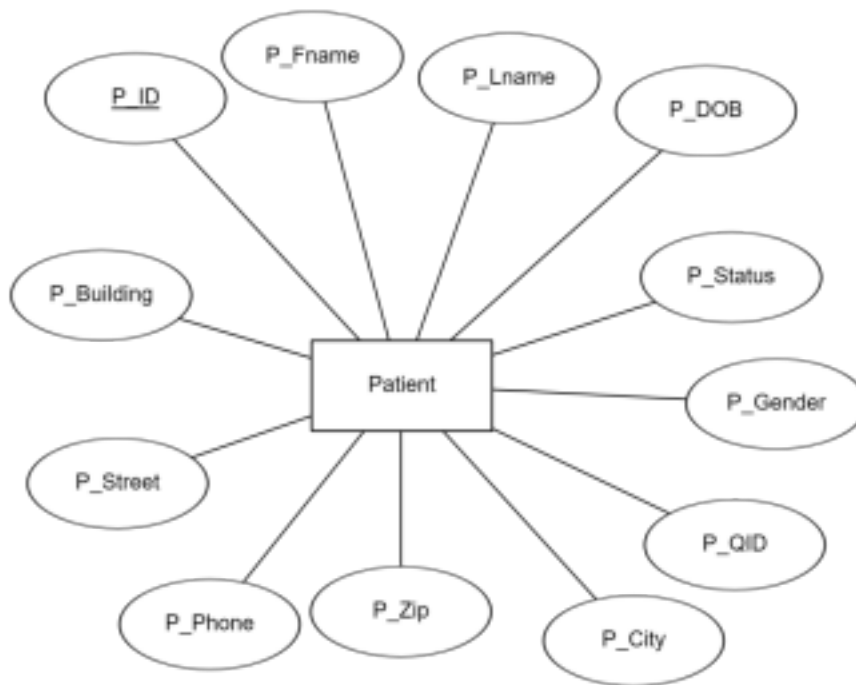


DA Project Report

4. TESTS

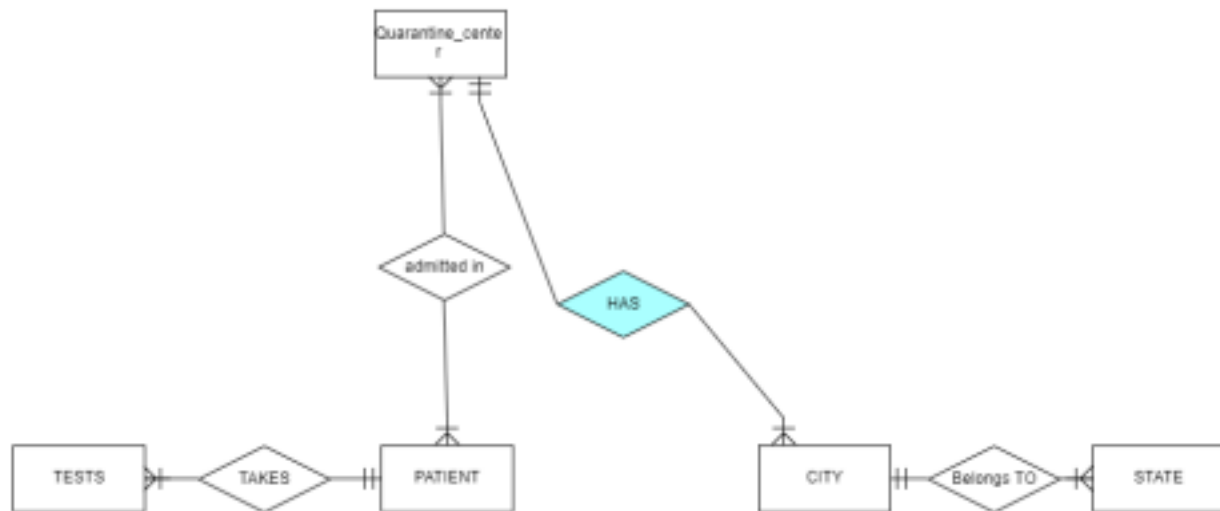


5. PATIENTS



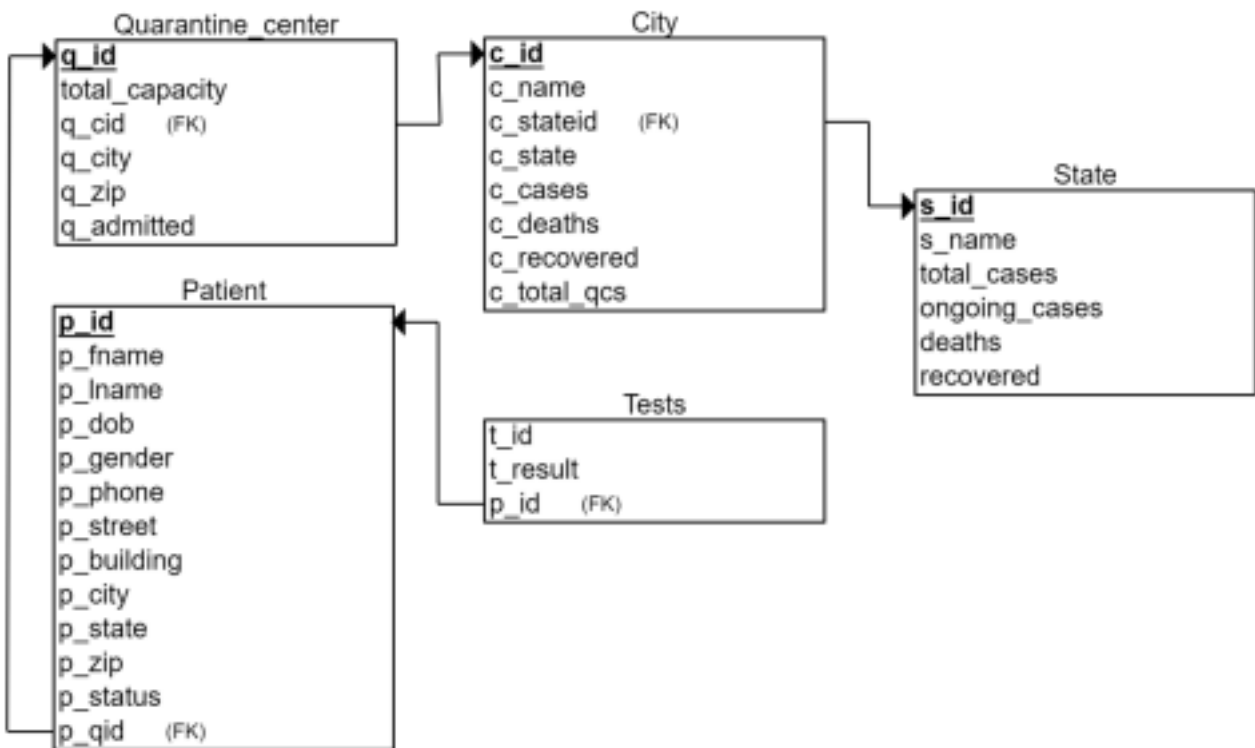
DA Project Report

ER_DIAGRAM



DA Project Report

Relational Model



DA Project Report

Normalization

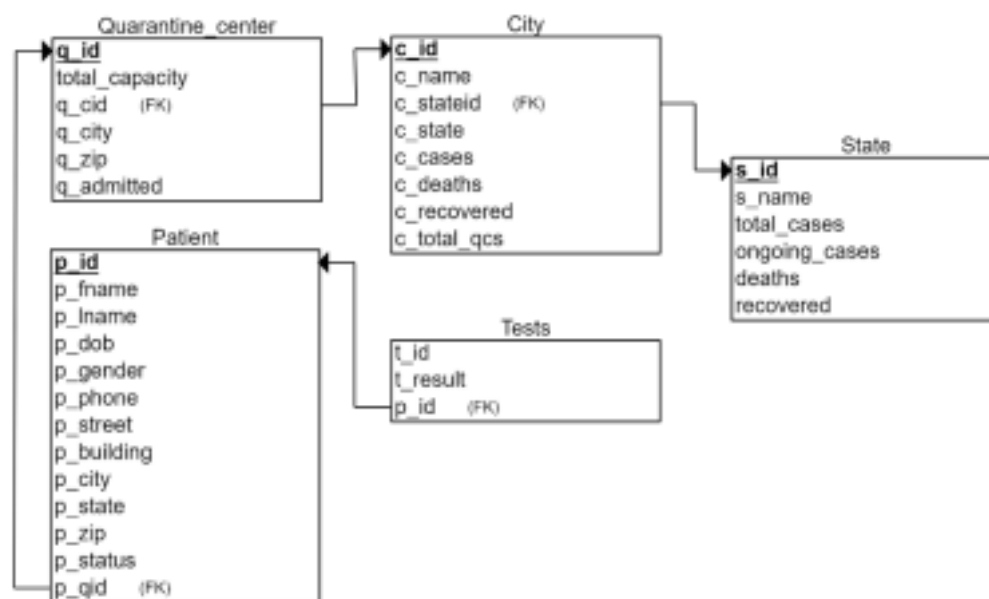
1. Normalization is the process of organizing the data in the database.
2. Normalization is used to minimize the redundancy from a relation or set of relations. It is also used to eliminate the undesirable characteristics like Insertion, Update and Deletion Anomalies.
3. Normalization divides the larger table into the smaller table and links them using relationship.
4. The normal form is used to reduce redundancy from the database table.

First Normal Form (1NF)

A relation will be 1NF if it contains an atomic value.

It states that an attribute of a table cannot hold multiple values. It must hold only single-valued attribute. First

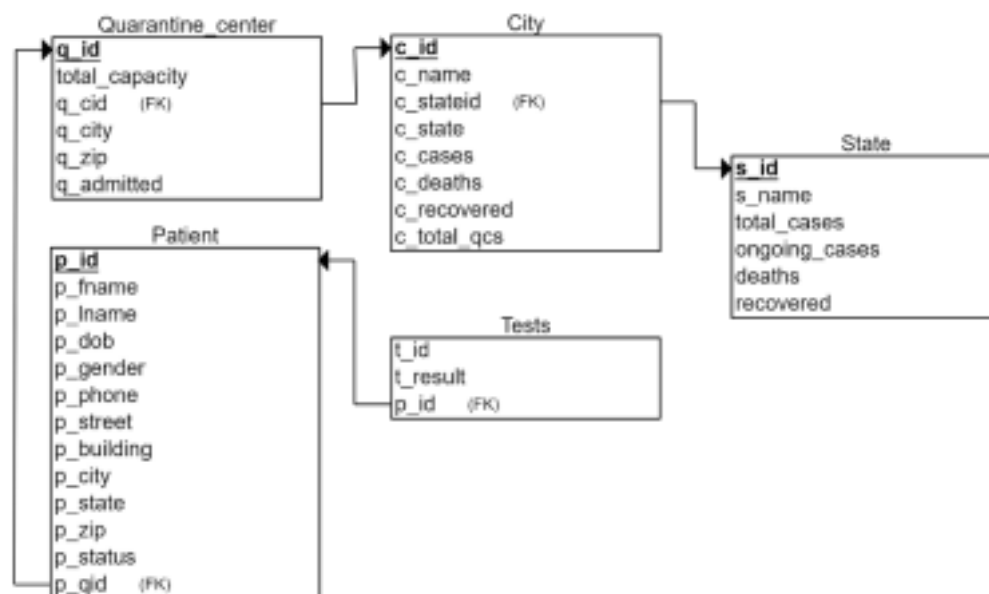
normal form disallows the multi-valued attribute, composite attribute, and their combinations. Our Tables already follows 1NF (we have tried to eliminate as much composite values and reduced the chances of 3 main anomalies occurring) .



Second Normal Form (2NF)

In the 2NF, relational must be in 1NF.

In the second normal form, all non-key attributes are fully functional dependent on the primary key. Our Table Follows 2nd Normal Form all of our non key attributes are functionally dependent on primary key DA Project Report



Third Normal Form (3NF)

A relation will be in 3NF if it is in 2NF and not contain any transitive partial dependency. 3NF is used to reduce the data duplication. It is also used to achieve the data integrity. If there is no transitive dependency for non-prime attributes, then the relation must be in third normal form. A relation is in third normal form if it holds atleast one of the following conditions for every non-trivial function dependency $X \rightarrow Y$.

X is a super key.

Y is a prime attribute, i.e., each element of Y is part of some candidate key.

Our Table does not follow 3rd Normal Form our elements do have transitive properties.

Boyce Codd normal form (BCNF)

BCNF is the advance version of 3NF. It is stricter than 3NF.

A table is in BCNF if every functional dependency $X \rightarrow Y$, X is the super key of the

table. For BCNF, the table should be in 3NF, and for every FD, LHS is super key.

Our Table does not follow 3rd Normal Form Therefore it does not follow BCNF as well

DA Project Report

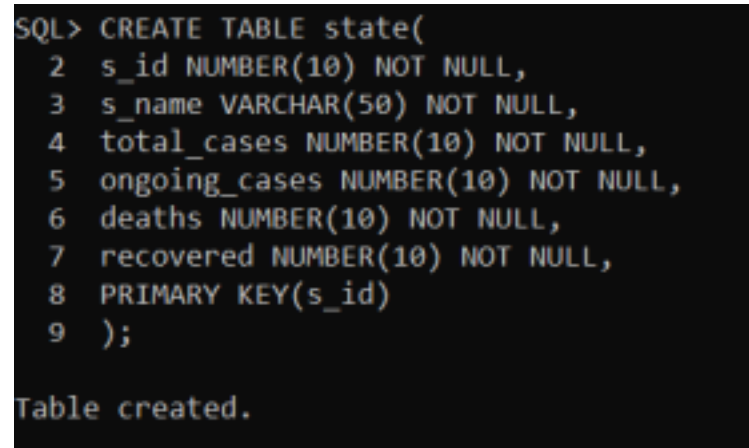
Create Statement

1. Creating STATE table

Command:

```
CREATE TABLE state(  
  s_id NUMBER(10) NOT NULL,  
  s_name VARCHAR(50) NOT NULL,  
  total_cases NUMBER(10) NOT NULL,  
  ongoing_cases NUMBER(10) NOT NULL,  
  deaths NUMBER(10) NOT NULL,  
  recovered NUMBER(10) NOT NULL,  
  PRIMARY KEY(s_id)  
);
```

Output:



```
SQL> CREATE TABLE state(  
  2  s_id NUMBER(10) NOT NULL,  
  3  s_name VARCHAR(50) NOT NULL,  
  4  total_cases NUMBER(10) NOT NULL,  
  5  ongoing_cases NUMBER(10) NOT NULL,  
  6  deaths NUMBER(10) NOT NULL,  
  7  recovered NUMBER(10) NOT NULL,  
  8  PRIMARY KEY(s_id)  
  9  );  
  
Table created.
```

2. Creating CITY table

Command:

```
CREATE TABLE city (  
  c_id NUMBER(10) PRIMARY KEY NOT NULL,  
  c_name VARCHAR(50) NOT NULL,
```

```
c_stateid NUMBER(10) NOT NULL,  
c_state VARCHAR(50) NOT NULL,  
c_cases NUMBER(10) NOT NULL,  
c_deaths NUMBER(10) NOT NULL,  
c_recovered NUMBER(10) NOT NULL,  
c_total_qcs NUMBER(10) NOT NULL,  
FOREIGN KEY(c_stateid) REFERENCES  
state(s_id) );
```

DA Project Report

Output:

```
SQL>  
SQL> CREATE TABLE city (  
2  c_id NUMBER(10) PRIMARY KEY NOT NULL,  
3  c_name VARCHAR(50) NOT NULL,  
4  c_stateid NUMBER(10) NOT NULL,  
5  c_state VARCHAR(50) NOT NULL,  
6  c_cases NUMBER(10) NOT NULL,  
7  c_deaths NUMBER(10) NOT NULL,  
8  c_recovered NUMBER(10) NOT NULL,  
9  c_total_qcs NUMBER(10) NOT NULL,  
10 FOREIGN KEY(c_stateid) REFERENCES state(s_id)  
11 );  
  
Table created.
```

3. Creating QUARANTINE_CENTER table

Command:

```
CREATE TABLE quarantine_center(  
q_id NUMBER(10) NOT NULL,  
total_capacity NUMBER(10) NOT NULL,  
q_name VARCHAR(50) NOT NULL,  
q_cid NUMBER(10) NOT NULL,  
q_city VARCHAR(50) NOT NULL,  
q_zip VARCHAR(50) NOT NULL,  
q_admitted NUMBER(10) NOT NULL,  
PRIMARY KEY(q_id),  
FOREIGN KEY(q_cid) REFERENCES city(c_id)  
);
```

DA Project Report

Output:


```
SQL>
SQL> CREATE TABLE quarantine_center(
  2  q_id NUMBER(10) NOT NULL,
  3  total_capacity NUMBER(10) NOT NULL,
  4  q_name VARCHAR(50) NOT NULL,
  5  q_cid NUMBER(10) NOT NULL,
  6  q_city VARCHAR(50) NOT NULL,
  7  q_zip VARCHAR(50) NOT NULL,
  8  q_admitted NUMBER(10) NOT NULL,
  9  PRIMARY KEY(q_id),
 10  FOREIGN KEY(q_cid) REFERENCES city(c_id)
 11 );

Table created.
```

4. Creating PATIENTS table

Command:

```
CREATE TABLE patient(
p_id NUMBER(10) PRIMARY KEY NOT NULL,
p_fname VARCHAR(50) NOT NULL,
p_lname VARCHAR(50) NOT NULL,
p_dob DATE NOT NULL,
p_gender VARCHAR(2) NOT NULL,
p_phone NUMBER(10) NOT NULL,
p_street VARCHAR(50) NOT NULL,
p_building VARCHAR(50) NOT NULL,
p_city VARCHAR(50) NOT NULL,
p_state VARCHAR(50) NOT NULL,
p_zip VARCHAR(50) NOT NULL,
p_status VARCHAR(50) NOT NULL,
p_qid NUMBER(10) NOT NULL,
FOREIGN KEY(p_qid) REFERENCES
quarantine_center(q_id) );
```

DA Project Report

Output:

```

SQL> CREATE TABLE patient(
  2  p_id NUMBER(10) PRIMARY KEY NOT NULL,
  3  p_fname VARCHAR(50) NOT NULL,
  4  p_lname VARCHAR(50) NOT NULL,
  5  p_dob DATE NOT NULL,
  6  p_gender VARCHAR(2) NOT NULL,
  7  p_phone NUMBER(10) NOT NULL,
  8  p_street VARCHAR(50) NOT NULL,
  9  p_building VARCHAR(50) NOT NULL,
 10  p_city VARCHAR(50) NOT NULL,
 11  p_state VARCHAR(50) NOT NULL,
 12  p_zip VARCHAR(50) NOT NULL,
 13  p_status VARCHAR(50) NOT NULL,
 14  p_qid NUMBER(10) NOT NULL,
 15  FOREIGN KEY(p_qid) REFERENCES quarantine_center(q_id)
 16 );

Table created.

```

5. Creating TESTS table

Command:

```

CREATE TABLE tests(
t_id NUMBER(10) NOT NULL,
t_result VARCHAR(20) NOT NULL,
p_id NUMBER(10) NOT NULL,
PRIMARY KEY(t_id),
FOREIGN KEY(p_id) REFERENCES patient(p_id)
);

```

Output:

```

SQL>
SQL> CREATE TABLE tests(
  2  t_id NUMBER(10) NOT NULL,
  3  t_result VARCHAR(20) NOT NULL,
  4  p_id NUMBER(10) NOT NULL,
  5  PRIMARY KEY(t_id),
  6  FOREIGN KEY(p_id) REFERENCES patient(p_id)
  7  );

Table created.

```

DA Project Report

Inserting Records into table

1. Inserting records into STATE table

Command:

```
INSERT INTO state VALUES  
(1,'Maharashtra',10000,3000,2000,1000); INSERT INTO state  
VALUES (2,'Delhi',15000,8000,4000,6000); INSERT INTO state  
VALUES (3,'Gujarat',5000,1000,2000,1000); INSERT INTO state  
VALUES (4,'Tamil Nadu',10000,3000,2000,1000);
```

Output:

S_ID	S_NAME	TOTAL_CASES	ONGOING_CASES	DEATHS	RECOVERED
1	Maharashtra	10000	3000	2000	1000
2	Delhi	15000	8000	4000	6000
3	Gujarat	5000	1000	2000	1000
4	Tamil Nadu	10000	3000	2000	1000

2. Inserting records into CITY table**Command:**

```
INSERT INTO city  
VALUES(10,'Mumbai',1,'Maharashtra',5000,3000,2000,300);  
INSERT INTO city  
VALUES(20,'Pune',1,'Maharashtra',3000,500,1000,100); INSERT INTO  
city VALUES(30,'Chennai',4,'Tamil  
Nadu',4000,1000,2000,200);  
INSERT INTO city  
VALUES(40,'Gurgoan',2,'Delhi',3000,1000,2000,300); INSERT INTO  
city VALUES(50,'Surat',3,'Gujarat',1500,500,1000,150);
```

Output:

DA Project Report



3. Inserting records into QUARANTINE_CENTER table

Command:

```
INSERT INTO quarantine_center VALUES (11,210,'Care
World',10,'Mumbai','401105',150);
INSERT INTO quarantine_center VALUES (12,234,'Hippy
Care',10,'Mumbai','401105',199);
INSERT INTO quarantine_center VALUES (13,154,'MG
Care',20,'Pune','424432',136);
INSERT INTO quarantine_center VALUES (14,100,'Diggy
Hospital',30,'Chennai','124314',78);
INSERT INTO quarantine_center VALUES
(15,130,'AIMS',40,'Gurgoan','546232',130);
INSERT INTO quarantine_center VALUES (16,211,'Victor
Health',30,'Chennai','423134',200);
INSERT INTO quarantine_center VALUES (17,120,'Plus
Medics',10,'Mumbai','401231',120);
INSERT INTO quarantine_center VALUES (18,90,'Cooper Care
Center',50,'Surat','424234',90);
```

Output:

DA Project Report



4. Inserting records into PATIENT table

Command:

```
INSERT INTO patient VALUES(1001,'Pranavi','Chintakindi',DATE
'1999-02- 07','F',9004901727,'Mg road','A
wing','Mumbai','Maharashtra','401105','ALIVE',11);
INSERT INTO patient VALUES(1002,'Sarathak','Karkera',DATE
'2000-05- 13','M',9104901345,'Sg road','Hinge
```

```

Plex','Mumbai','Maharashtra','401105','ALIVE',12);
INSERT INTO patient VALUES(1003,'Manan','Kataria',DATE
'1999-03- 17','M',9866741123,'Cg road','Home
Care','Pune','Maharashtra','401105','ALIVE',13);
INSERT INTO patient VALUES(1004,'Akshat','Vaishya',DATE
'1999-04- 27','F',9864564565,'FP road','Drogo
Building','Chennai','Tamil Nadu','124314','ALIVE',16);
INSERT INTO patient VALUES(1005,'Harshi','Rajput',DATE
'2000-08- 09','F',9908764043,'Magi road','Sandy
Apts','Gurgoan','Delhi','642313','DEAD',15);
INSERT INTO patient VALUES(1006,'Priyanka','Choudhary',DATE
'1997-03- 27','F',9127658745,'Santa road','Hieghted
Skies','Surat','Gujarat','234142','ALIVE',18);
INSERT INTO patient VALUES(1007,'Sanya','Malhotra',DATE
'1992-10- 17','F',9834556732,'Sea road','Light
Flats','Chennai','Tamil
Nadu','423134','DEAD',14);
INSERT INTO patient VALUES(1008,'Manveer','Singh',DATE
'1995-12- 25','M',9965423762,'Diva road','Apex
Heights','Surat','Gujarat','424234','DEAD',18);

```

DA Project Report

```

INSERT INTO patient VALUES(1009,'Pooja','Reddy',DATE '1988-12-
09','F',9953467543,'Gandhi road','Hints
Complex','Gurgoan','Delhi','546232','ALIVE',15);
INSERT INTO patient VALUES(1010,'Pradnya','Rane',DATE
'1999-04- 01','F',9954653287,'Powder road','89 Miling
Apts','Pune','Maharashtra','424432','DEAD',13);
INSERT INTO patient VALUES(1011,'Sneha','Reddy',DATE '1997-05-
13','F',9002345727,'Diner Street','Bings
Apts','Mumbai','Maharashtra','401102','DEAD',11);
INSERT INTO patient VALUES(1012,'Vijay','Khanna',DATE
'2000-05- 13','M',9104901345,'Pitty road','Hinge
Plex','Mumbai','Maharashtra','403423','DEAD',12);
INSERT INTO patient VALUES(1013,'Shalini','Singh',DATE
'1999-02- 07','F',9012341727,'Fink road','Bing
wings','Mumbai','Maharashtra','405453','ALIVE',17);
INSERT INTO patient VALUES(1014,'Pravin','Pandey',DATE
'2000-05- 13','M',9101342145,'HG Marg','Hinge
Apts','Mumbai','Maharashtra','401311','ALIVE',12);
INSERT INTO patient VALUES(1015,'Sharif','Khan',DATE '1999-02-
07','F',9004901727,'Puny Road','Apt
Life','Mumbai','Maharashtra','401312','ALIVE',17);
INSERT INTO patient VALUES(1016,'Aman','Singh',DATE '2000-05-
13','M',9167867345,'Hint Border','Locks
Apts','Mumbai','Maharashtra','401231','ALIVE',11);

```

Output:



DA Project Report

5. Inserting records into TESTS table

Command:

```
INSERT INTO tests
VALUES(101,'POSITIVE',1001); INSERT INTO
tests VALUES(102,'NEGATIVE',1002); INSERT
INTO tests VALUES(103,'POSITIVE',1003);
INSERT INTO tests
VALUES(104,'NEGATIVE',1004); INSERT INTO
tests VALUES(105,'POSITIVE',1005); INSERT
INTO tests VALUES(106,'POSITIVE',1006);
INSERT INTO tests
VALUES(107,'POSITIVE',1007); INSERT INTO
tests VALUES(108,'POSITIVE',1008); INSERT
INTO tests VALUES(109,'POSITIVE',1009);
INSERT INTO tests
VALUES(110,'POSITIVE',1010); INSERT INTO
tests VALUES(111,'POSITIVE',1011); INSERT
INTO tests VALUES(112,'POSITIVE',1012);
INSERT INTO tests
VALUES(113,'NEGATIVE',1013); INSERT INTO
tests VALUES(114,'NEGATIVE',1014); INSERT
INTO tests VALUES(115,'NEGATIVE',1015);
INSERT INTO tests
VALUES(116,'POSITIVE',1016);
```

Output:



DA Project Report

SELECT Commands

Multiple SELECT functions

Description:

CONCAT: Adds two or more strings together

ALIAS: Aliases are often used to make column names more readable. An alias only exists for the duration of that query. An alias is created with the AS keyword.

Command:

```
SELECT p_id, CONCAT (p_fname, p_lname) AS p_name, p_building || ' ' ||  
p_street || ' ' || p_zip AS p_address FROM patient;
```

Output:



DA Project Report

Conditional SELECT statements:

Command:

```
SELECT p.p_id,p.p_fname || ' ' || p.lname AS p_name, p_status FROM patient p
WHERE p.p_status = 'ALIVE';
```

Output:



DA Project Report

Selecting Multiple tables using Conditional statements

Command:

```
SELECT p.p_id,p.p_fname || ' ' || p.lname AS p_name, t.t_id, t.t_result FROM  
tests t, patient p WHERE t_result = 'POSITIVE' and p.p_id = t.p_id;
```

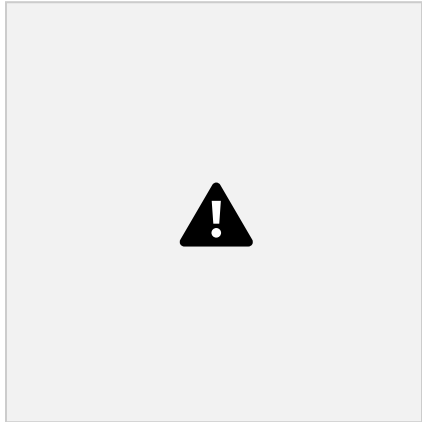
Output:



Selecting records using GROUP BY clause

Command:

```
SELECT q_city,COUNT(q.q_id) FROM quarantine_center q GROUP BY  
q_city; Output:
```



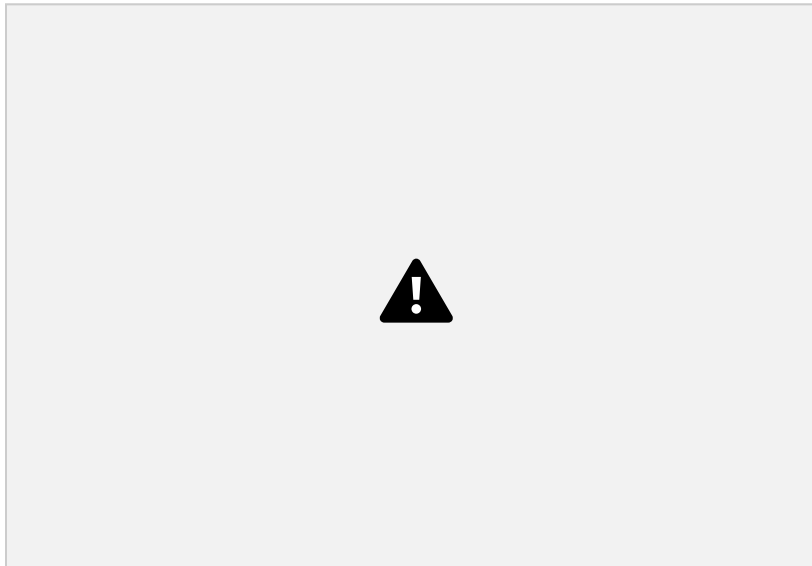
DA Project Report

Selecting records using ORDER BY clause

Command:

```
select q.q_id,q.q_name,p.p_id,p.p_fname || ' ' ||p_lname as p_name, p_status  
from quarantine_center q ,patient p  
where p.p_status='ALIVE' and p.p_qid = q.q_id ORDER BY q.q_id;
```

Output:

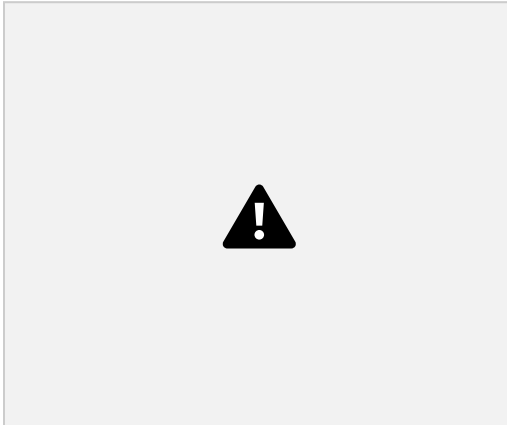


Selecting records using ORDER BY and GROUP BY clause

Command:

```
select q_city,count(q.q_id) as Quarantine_Centers from quarantine_center q group by  
q_city order by q_city asc;
```

Output:



DA Project Report

Updating Records

Command:

```
UPDATE tests SET t_result = 'POSITIVE' WHERE p_id =  
1005; Output:
```

Before Update:



After Update



Altering table

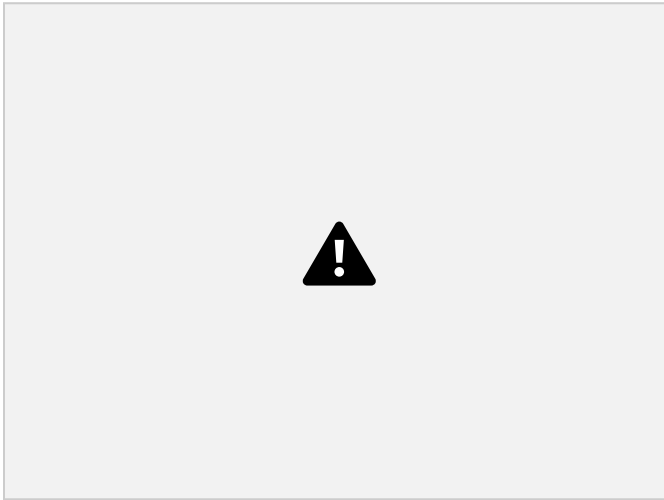
Modify Table

Command:

```
ALTER TABLE state RENAME COLUMN ONGOING_CASES TO
```

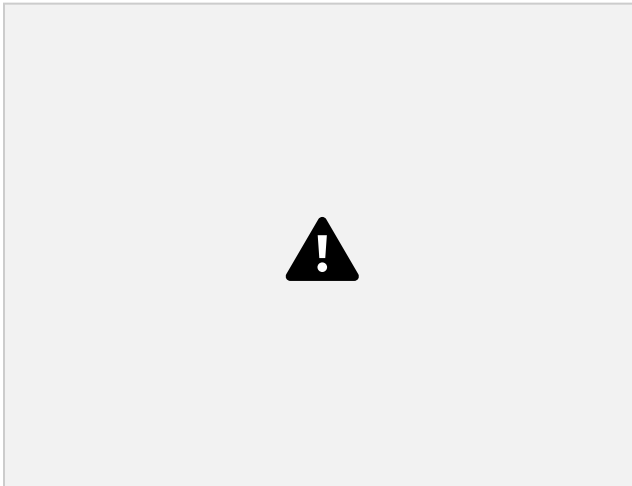
ONGOING; **Output:**

Before Alter



DA Project Report

After Alter



[Rollback](#)

Description: Undos the transactions that have not been saved in the database

Command:

`rollback;`

Output:



DA Project Report

Creating View Tables

Description: In SQL, a view is a virtual table based on the result-set of an SQL statement. A view contains rows and columns, just like a real table. The fields in a view are fields from one or more real tables in the database. You can add SQL statements and functions to a view and present the data as if the data were coming from one single table.

Command:

```
create view city_1 as select * from city;
create view state_1 as select * from state;
create view patient_1 as select * from patient;
create view quarantine_1 as select * from
quarantine_center; create view tests_1 as select * from
tests;
```

Output:



DA Project Report

Joins

1. Cross Join

Command:

```
SELECT p.p_id,p.p_fname || ' ' || p.lname as p_name,t.t_id,t.t_result  
FROM patient p CROSS JOIN tests t WHERE p.p_id = t.p_id AND  
t.t_result='POSITIVE';
```

Output:



2. Joining three tables

Command:

```
SELECT c.c_id, c.c_name,s.s_name, q.q_name FROM state s, city c,  
quarantine_center q WHERE s.s_id = c.c_stateid AND c.c_id = q.q_cid  
order BY c.c_id;
```

DA Project Report

Output:



3. Full Outer Join

Command:

```
SELECT p_id, p_fname || ' ' || p_lname as p_name, q_name,  
q_id FROM patient  
FULL OUTER JOIN quarantine_center  
ON p_qid = q_id  
WHERE p_status='ALIVE';
```

Output:



It is important for every element in a table to have a purpose and it should not have any redundancy or inaccuracy this is why it is very important for a database to be very well planned and robust and we have hoped to have achieved that here.