Pune Vidyarthi Griha's

College of Engineering and Technology & G. K. Pate (Wani) Institute of Management

Approved by AICTE, DTE (Code: 6274) | Affiliated to SPPU, Pune | NAAC Second Cycle 'A' Grade

Database Management System
Mini project
On

"Oxygen Supply Management System"

Submitted to the

Savitribai Phule Pune University
In partial fulfilment for the award of the Degree of
Bachelor of Engineering
in
Information Technology

By

S190078572 Neha Sonar
S190078538 Priyanka Kothawade
S190078508 Komal Badhe
S190078571 Tushar Shinde

Under the guidance of
Prof. N. R. Sonawane

Department of Information Technology
PVGCOET & GKPIM, Pune
Pune, Maharashtra 411009
2020-2021

CERTIFICATE

This is to certify that the mini project report entitled **"Oxygen Supply Management System"** being submitted by Neha Sonar(S190078572), Priyanka Kothawade(S190078538), Komal Badhe(S190078508), Tushar Shinde(S190078571) is a record of bonafide work carried out by him/her under the supervision and guidance of Prof.N.R.Sonawane in partial fulfillment of the requirement for **SE (Information Technology Engineering) – 2019 course** of Savitribai Phule Pune University, Pune in the academic year 2020-2021.

Date: 31.05.2021

Place: Pune

Prof.N.R.Sonawane	Prof. S.A. Mahajan
Project Guide	Head of the Department

Dr. Manoj R. Tarambale

Principal

This Mini Project report has been examined by us as per the Savitribai Phule Pune University, Pune requirements at PVG's College of Engineering and Technology & G. K. Pate (Wani) Institute of Management, Pune-09 on 31.05.2021

Internal Examiner External Examiner

ACKNOWLEDGEMENT

This major project would not have been possible without the valuable assistance of many people to whom we are indebted, in particular, our project guide prof. N. R. Sonawane of Pune Vidyarthi Griha’s College of Engineering. We would also like to thank “Department of Information Technology”, for providing us necessary components for project. Our thank also goes to all the teachers of Information Technology Department who helped us in many difficult situations regarding the project and provided with the necessary advice. A special word of thanks is to our class mates and our families for providing us the moral support.

(Students Name & Signature)

CONTENTS

1. ABSTRACT

In this project we have created one application which is easy to access user friendly. For this application we used the backend as Mysql to store the data which is used in the application. Two kinds of people are able to use this application as supplier and the hospital(consumer) as well. The hospital is able to placed order for oxygen cylinder.

Harrowing stories of patients dying for lack of oxygen in Maharashtra hospitals have focused international attention on where this life-sustaining gas comes from. Oxygen is all around us in the air we breathe, but when our lungs become inflamed and damaged, as they do during severe coronavirus infections, it makes it harder for them to absorb enough oxygen to meet the body’s needs. All of our cells need oxygen to survive, and as blood oxygen levels begin to fall, our organs and tissues begin to starve. To combat this, patients hospitalised with severe COVID-19 are often given supplemental oxygen.

As coronavirus cases in India have soared, medical oxygen consumption has risen eight-fold, and dozens of hospitals in Delhi and elsewhere have reported running out of the supplies they need to keep patients alive. Various countries have responded with offers of oxygen cylinders, concentrators and cryogenic tanks to transport ultra-cold liquid oxygen, while organisations such as UNICEF are working with their partners to buy and install oxygen generation plants in hospitals.

2. INTRODUCATION

In this project we developed an oxygen supply management system.

Basically, this project is created using MySQL database. It contains 4 entities which are as follows:

- Oxygen_plant
- Supplier
- Hospital
- Order

All the entities have relation between each other as follows :

1. Oxygen_plant supply oxygen cylinder to Suppliers:
   One oxygen plant can supply to many suppliers.

2. Supplier supply Oxygen cylinder to Hospital:
   Many suppliers can supply oxygen cylinders to many hospitals.

3. Hospital Places Orders:
   Many hospitals can place many orders of oxygen cylinders.

4. Supplier Places Orders:
   Many suppliers can place many orders of oxygen cylinders.

With this schema our database can perform following operations:

- Keep the record of available oxygen.
- Provide the accurate amount of oxygen to the hospitals.
- Provide oxygen in various cities with the help of suppliers.
- Keep record of supplied oxygen.
- From the above information make the presumption to make required oxygen cylinders.

History of SQL (A Research Project Conducted by IBM) :

The origins of the SQL language date back to a research project conducted by IBM at their research laboratories in San Jose, California in the early 1970s. The aim of the project was to develop an experimental RDBMS which would eventually lead to a marketable product. At that time, there was a lot of interest in the relational model for databases at the academic level, in conferences and seminars. IBM, which already had a large share of the commercial database market with hierarchical and network model DBMSs, realized quite quickly that the relational model would figure prominently in future database products. The project at IBM's San Jose labs was started in 1974 and was named System R. A language called Sequel (for Structured English Query Language) was chosen as the relational database language for System R. In the project, Sequel was abbreviated to SQL. This is the reason why SQL is still generally pronounced as see-quell. In the first phase of the System R project, researchers concentrated on developing a basic version of the RDBMS. The main aim at this stage was to verify that the theories of the relational model could be translated into a working, commercially viable product. This first phase was successfully completed by the end of 1975, and resulted in a rudimentary, single-user DBMS based on the relational model. The subsequent phases of System R concentrated on further

developing the DBMS from the first phase. Additional features were added, multi-user capability was implemented, and by 1978, a completed RDBMS was ready for user evaluation. The System R project was finally completed in 1979. During this time, the SQL language was modified and added to as the needs of the System R DBMS dictated. During the development of System R and SQL/DS, other companies were also at work creating their own relational database management systems. Some of them, Oracle being a prime example, even implemented SQL as the relational database language for their DBMSs concurrently with IBM. Today, the SQL language has gained ANSI (American National Standards Institute) and ISO (International Standards Organization) certification. A version of SQL is available for almost any hardware platform from CRAY supercomputers to IBM PC microcomputers. In recent years, there has been a marked trend for software manufacturers to move away from proprietary database languages and settle on the SQL standard. The microcomputer platform especially has seen a proliferation of previously proprietary packages that have implemented SQL functionality. Even spreadsheet and word processing packages have added options which allow data to be sent to and retrieved from SQL based databases via a Local Area or a Wide Area network connection.[3].

a. Problem Statement :

Oxygen Supply Management System.

b. Motivation :

- As we faced many oxygen requirement issues due to corona.
- Many corona patients died due to unavailability of oxygen. So, with the help of this project we can able to calculate our daily oxygen requirement. Also, hospitals can able to order oxygen cylinders through many suppliers so that the needy patient get the oxygen supply in time. So, our motivation is this current pandemic covid situation.

c. Objectives:

- To learn and understand various Database Architectures and its use for application development.
- Understand the fundamental concepts of database management. These concepts include aspects of database design, database languages, and database-system implementation.
- To give systematic database design approaches covering conceptual design, logical design and an overview of physical design.
- To learn the SQL database system.
- To learn and understand various Database Architectures and its use for application development.
- To program PL/SQL including stored procedures, stored functions, cursors and packages.

3. Data Type

SQL Numeric Data Types :

Datatype	From	To
bit	0	1
tinyint	0	255
smallint	-32,768	32,767
int	-2,147,483,648	2,147,483,647
bigint	-9,223,372,036, 854,775,808	9,223,372,036, 854,775,807
decimal	-10^38 +1	10^38 -1
numeric	-10^38 +1	10^38 -1
float	-1.79E + 308	1.79E + 308
real	-3.40E + 38	3.40E + 38

SQL Date and Time Data Types:

Datatype	Description
DATE	Stores date in the format YYYY-MM-DD
TIME	Stores time in the format HH:MI:SS
DATETIME	Stores date and time information in the format YYYY-MM-DD HH:MI:SS
TIMESTAMP	Stores number of seconds passed since the Unix epoch (‘1970-01-01 00:00:00’ UTC)
YEAR	Stores year in 2 digit or 4 digit format. Range 1901 to 2155 in 4-digit format. Range 70 to 69, representing 1970 to 2069.

SQL Character and String Data Types :

Datatype	Description
CHAR	Fixed length with maximum length of 8,000 characters
VARCHAR	Variable length storage with maximum length of 8,000 characters
VARCHAR(max)	Variable length storage with provided max characters, not supported in MySQL
TEXT	Variable length storage with maximum size of 2GB data

Note: All the above data types are for character stream, they should not be used with Unicode data.

SQL Unicode Character and String Data Types :

Datatype	Description
NCHAR	Fixed length with maximum length of 4,000 characters
NVARCHAR	Variable length storage with maximum length of 4,000 characters
NVARCHAR(max)	Variable length storage with provided max characters
NTEXT	Variable length storage with maximum size of 1GB data

4. Data Requirement

REQUIREMENTS COLLECTION AND ANALYSIS :

We list the data requirements for the database project here, and then create its conceptual schema step-by-step as we introduce the modeling concepts of the ER model. The MH_Oxygen_Plants database keeps track on oxygen supply, suppliers, and orders. Suppose that after the requirements collection and analysis phase, the database designers provide the following description of the mini world the part of the Oxygen_Plant that will be represented in the database.

The MH_Oxygen_Plants database is organized into Hospitals. Each Hospital has a unique name, a unique id, address, pin code and a particular supplier who provides oxygen to the hospital. We keep track of the orders when the hospital places order for oxygen cylinder then particular supplier assigned to that hospital will provide them required amount of oxygen cylinders from oxygen plant. The suppliers are from many cities in Maharashtra.

a. A Oxygen_plant keeps track on daily production on oxygen, daily supplied oxygen, on suppliers and plant status.

b. In suppliers, every supplier has his unique id and contact number. Suppliers are assigned with the hospitals in Maharashtra and keeps track on order table. When the order will be placed by particular hospital then supplier will take that order and provide them from oxygen plant. Also, display the status about it.

Entity Types, Entity Sets, Attributes, and Keys:

1. **Oxygen_Plant :**

Attributes: Plant_id, Plant_name, Daily_capacity, Daily_supply, Total_supplier, Contact_no and Plant_status.

Keys: We specified Plant_id as primary key.

2. Supplier :

Attributes: Supplier_id, Supplier_name, Hospital_name, Contact_no, Supply_city, Hospital_count, Available_oxygen, Supplied_oxygen, Supplier_status.

Keys: We specified Supplier_id as primary key.

3. Hospital :

Attributes: Hospital_id, Hosp_Address, Pin_code, Total_patients, Require_oxygen, Supplier_id.

Keys: We specified Hospital_id as primary key.

And, Supplier_id as Foreign key.

4. Order :

Attributes: Order_id, Supplier_id, Hospital_id, Order_status.

Keys: We specified Order_id as primary key.

And, Supplier_id and Hospital_id as Foreign keys.

5. E-R Diagram

An entity–relationship model (ER model) describes inter-related things of interest in a specific domain of knowledge. An ER model is composed of entity types and specifies relationships that can exist between instances of those entitytypes.

In software engineering an ER model is commonly formed to represent things that abusiness needs to remember in order to perform business processes. Consequently, the ER model becomes an abstract data model that defines a data or information structure that can beimplemented in a database, typically a relational database.

Entity-relationship modeling was developed for database design by Peter Chen and published in a 1976 paper. However, variantsof the idea existed previously, some ER modelers show super and subtype entities connected by generalization-specialization relationships, and an ER model can be used also in the specification of domain-specific ontology.

An ER model is typically implemented as a database. In a simple relational database implementation, each row of a table represents one instance of an entity type and each field in a table represents an attribute type. In a relational database a relationship between entities is implemented by storing the primary key of one entity as a pointer or foreign key in the table of another entity. There is a tradition for ER/data models tobe built at two or three levels of abstraction.

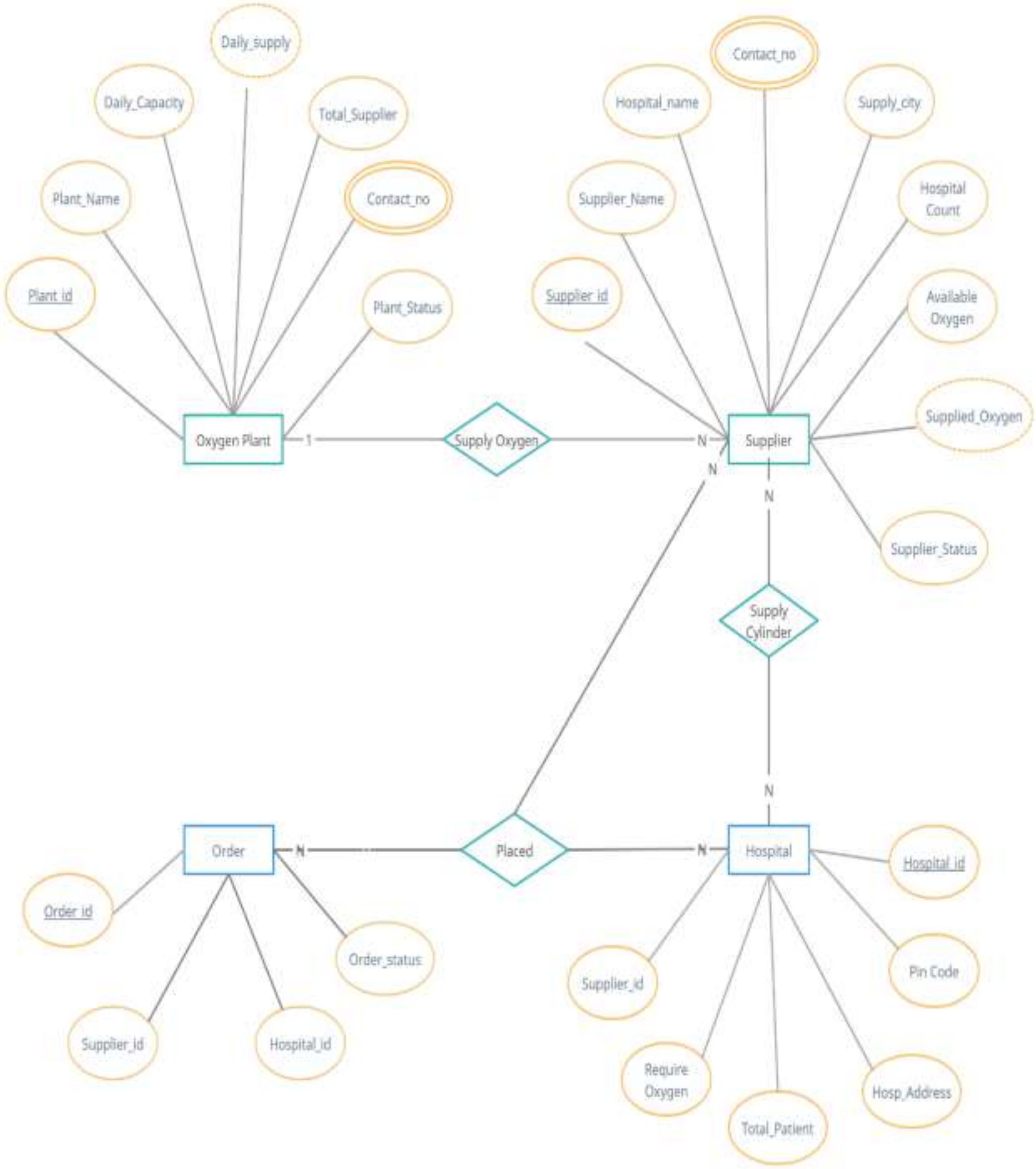


Figure: The ER conceptual schema diagram for the MH_Oxygen_Plants database.

6. Schema Diagram

A database schema is the skeleton structure that represents the logical view of the entire database. It defines how the data is organized and how the relations among them are associated. It formulates all the constraints that are to be applied on the data.

A database schema defines its entities and the relationship among them. It contains a descriptive detail of the database, which can be depicted by means of schema diagrams. It’s the database designers who design the schema to help programmers understand the database and make it useful.

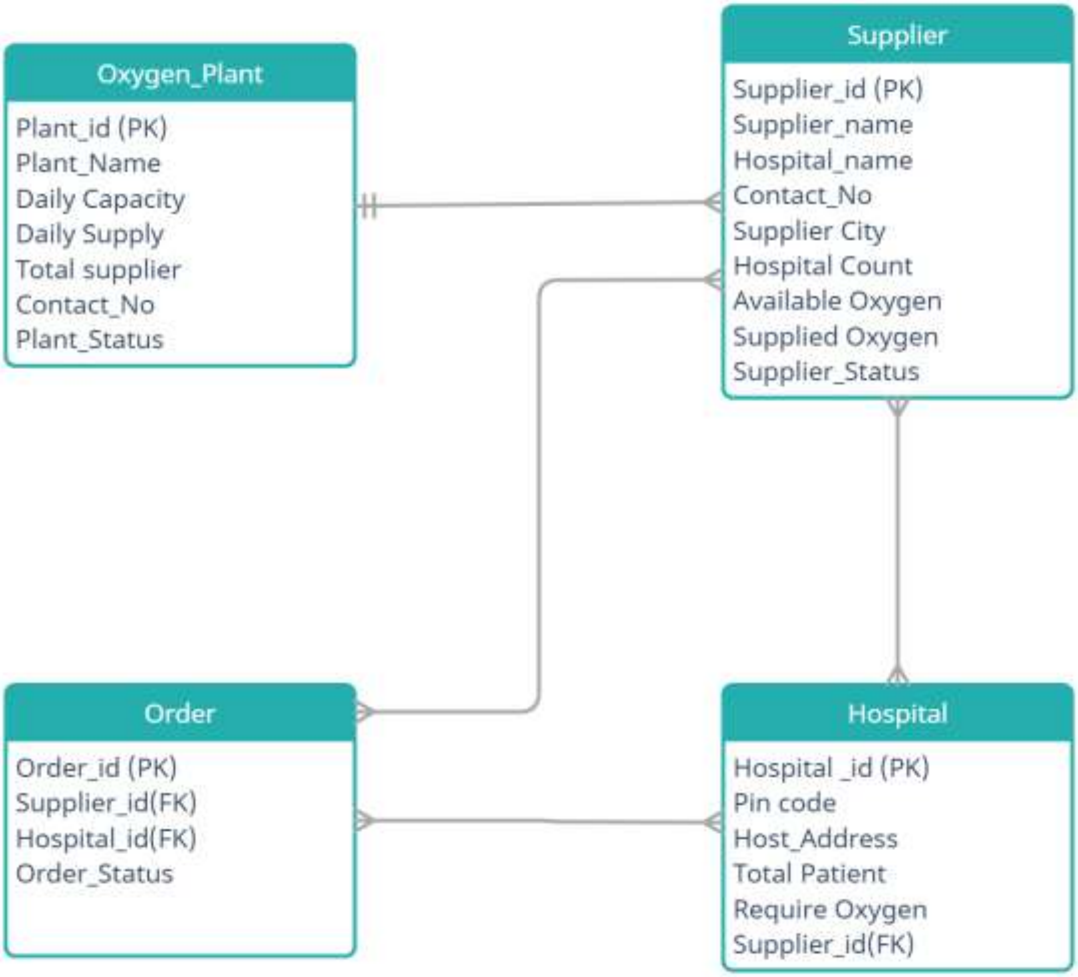


Figure : Schema Diagram

7. Relational Database Design

The MH_Oxygen_Plants schema is shown again in above figure and the corresponding MH_Oxygen_Plants relational database schema is shown in Figure.

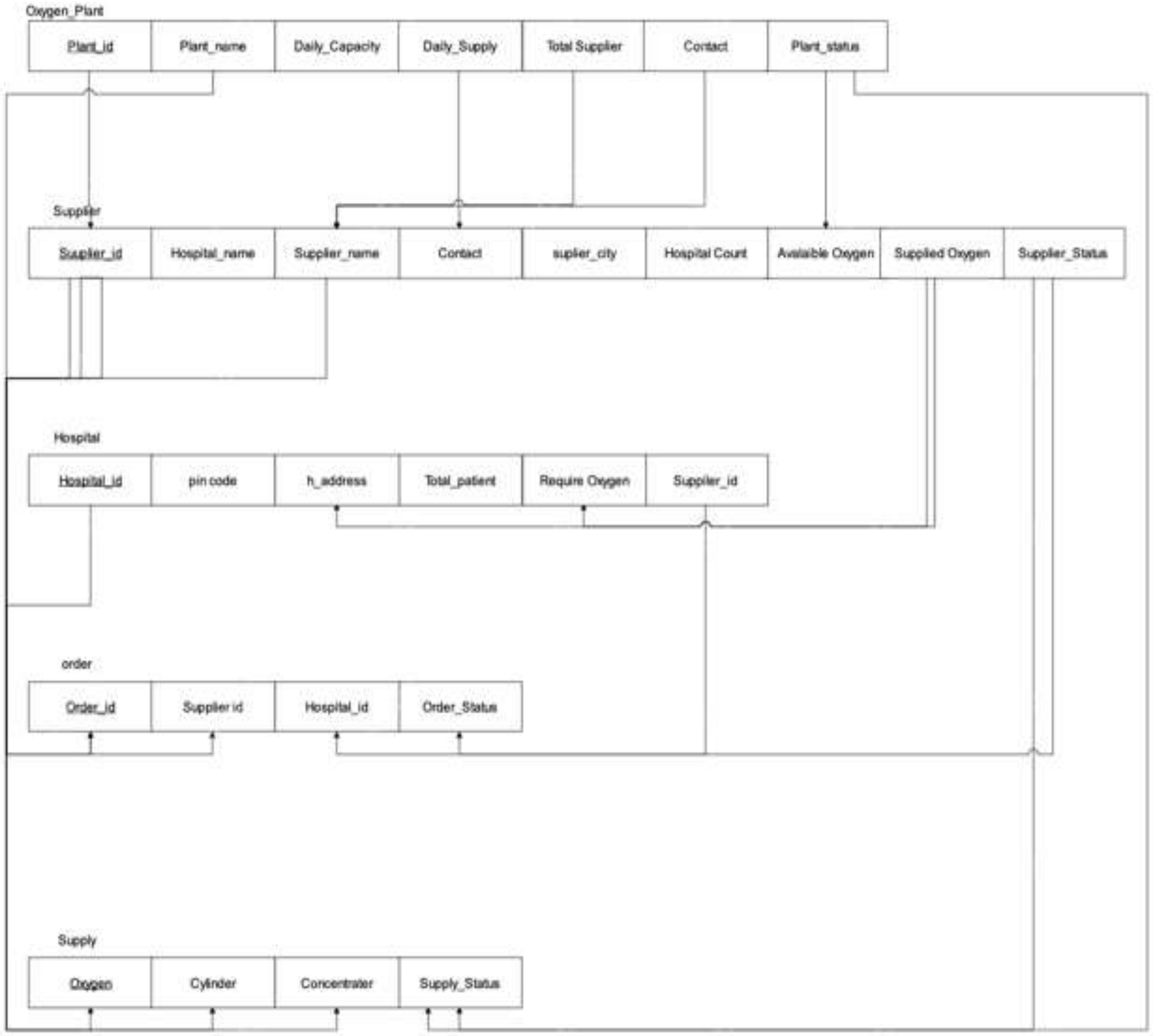


Figure : Relational Database Design

ER-to-Relational Mapping Algorithm :

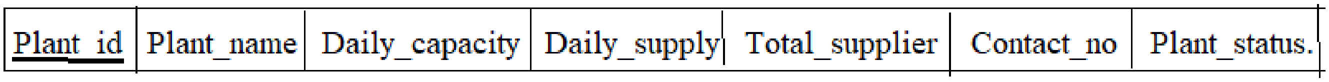
The relational model constraints, which include primary keys, unique keys (if any), and referential integrity constraints on the relations, will also be specified in the mapping results.

Step 1: Mapping of Regular Entity Types

- For each regular/strong entity type, create a corresponding relation that includes all the simple attributes (includes simple attributes of composite relations)
- Choose one of the key attributes as primary § If composite, the simple attributes together form the primary key
- Any remaining key attributes are kept as secondary unique keys (these will be useful for physical tuning w.r.t. indexing analysis)

Step 1 result:

Oxygen_plant



Plant_id	Plant_name	Daily_capacity	Daily_supply	Total_supplier	Contact_no	Plant_status.

Supplier

Sup_id	Sup_name	H_name	Contact_no	Sup_city	H_count	A_oxygen	S_oxygen	Sup_status

Hospital

Hospital_id	Hosp_Address	Pin_code	Total_patients	Require_oxygen	Supplier_id.

Orders

Order_id	Supplier_id	Hospital_id	Order_status

Step 2: Mapping of Weak Entity Types.

- For each weak entity type, create a corresponding relation that includes all the simple attributes
- Add as a foreign key all of the primary key attribute(s) in the entity corresponding to the owner entity type
- The primary key is the combination of all the primary key attributes from the owner and the partial key of the weak entity, if any

Note: Dark line indicates primary key.

Step 2 result:

Oxygen_plant

Plant_id	Plant_name	Daily_capacity	Daily_supply	Total_supplier	Contact_no	Plant_status.

Supplier

Sup_id	Sup_name	H_name	Contact_no	Sup_city	H_count	A_oxygen	S_oxygen	Sup_status

Hospital

Hospital_id	Hosp_Address	Pin_code	Total_patients	Require_oxygen	Supplier_id.

Orders

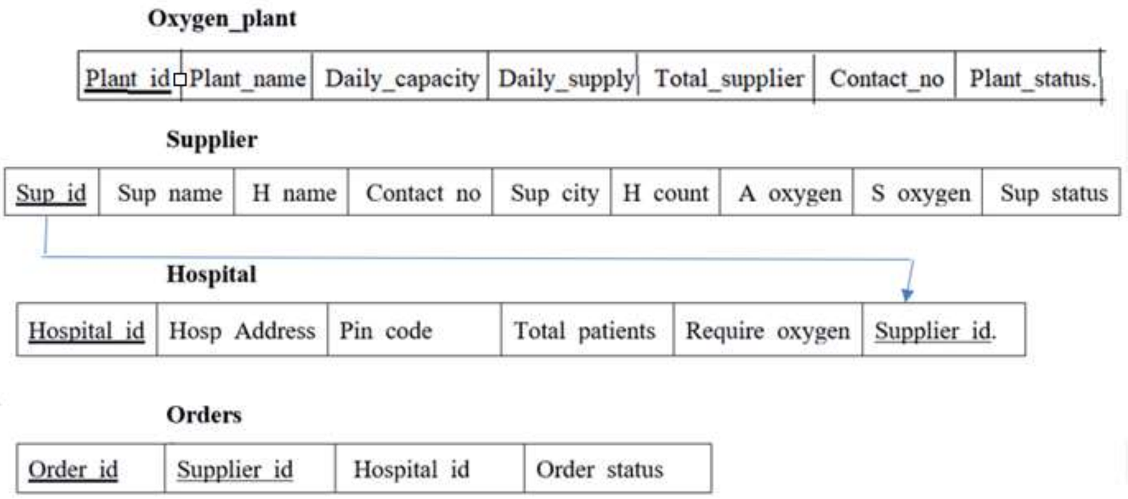
Order_id	Supplier_id	Hospital_id	Order_status

Step 3: Mapping of Binary 1:1 Relationship Types.

- Choose one relation as S, the other T Better if S has total participation (reduces number of NULL values)
- Add to S all the simple attributes of the relationship
- Add as a foreign key in S the primary key attributes of T

Step 3 result:

1. Foreign key approach:



Oxygen_plant

Plant_id	Plant_name	Daily_capacity	Daily_supply	Total_supplier	Contact_no	Plant_status.

Supplier

Sup_id	Sup_name	H_name	Contact_no	Sup_city	H_count	A_oxygen	S_oxygen	Sup_status

Hospital

Hospital_id	Hosp_Address	Pin_code	Total_patients	Require_oxygen	Supplier_id.

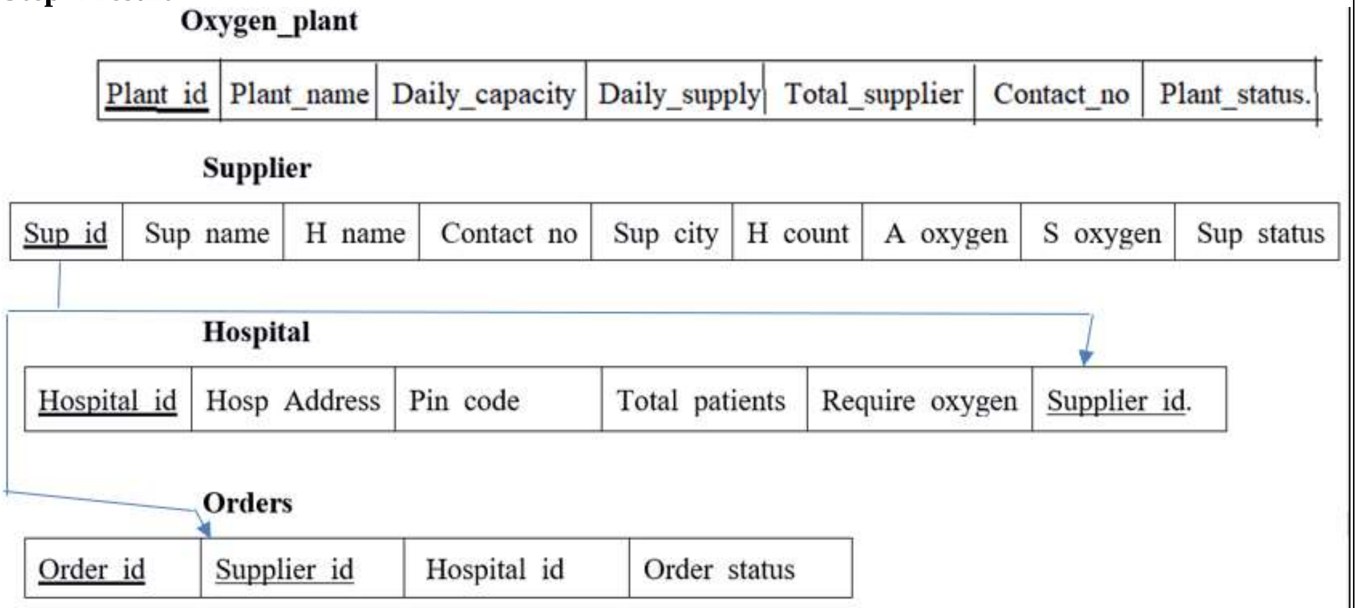
Orders

Order_id	Supplier_id	Hospital_id	Order_status

Step 4: Mapping of Binary 1:N Relationship Types.

- Choose the S relation as the type at the N-side of the relationship, other is T
- Add as a foreign key to S all of the primary key attribute(s) of T

Step 4 result



Oxygen_plant

Plant_id	Plant_name	Daily_capacity	Daily_supply	Total_supplier	Contact_no	Plant_status.

Supplier

Sup_id	Sup_name	H_name	Contact_no	Sup_city	H_count	A_oxygen	S_oxygen	Sup_status

Hospital

Hospital_id	Hosp_Address	Pin_code	Total_patients	Require_oxygen	Supplier_id.

Orders

Order_id	Supplier_id	Hospital_id	Order_status

Step 5 : Mapping of Multivalued Attributes.

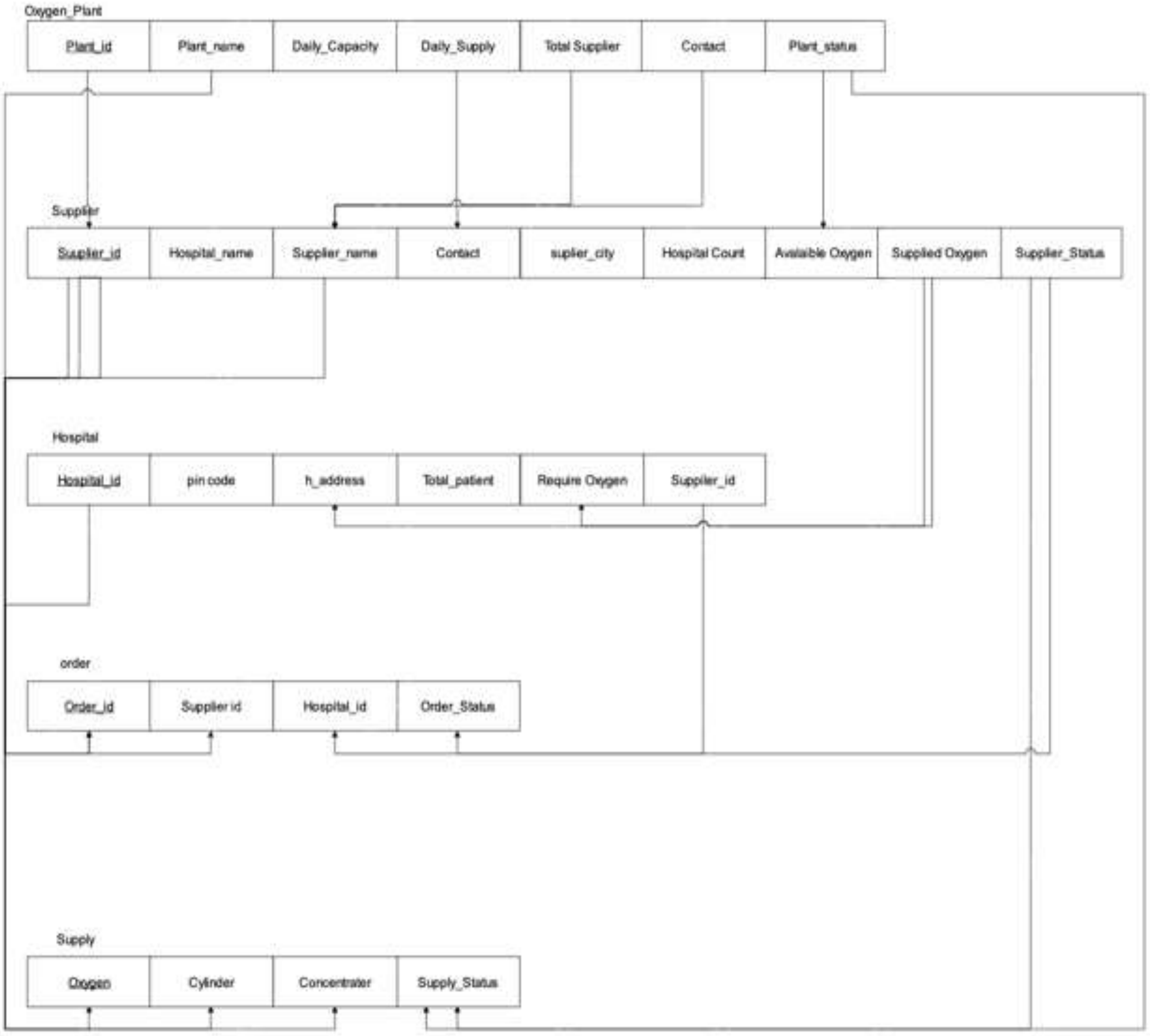
- Create a new relation S
- Add as foreign keys the primary keys of the corresponding relation
- Add the attribute to S (if composite, the simple attributes); the combination of all attributes in S forms the primary key

  Note : The relation in oxygen plant database does not contain any multivalued attribute, hence this step is not applicable.

Step 6: Mapping of Binary M:N Relationship Types.

- Create a new relation S (termed: relationship relation) – In some ERD dialects, actually drawn in Add as foreign keys the primary keys of both relations; their combination forms the primary key of S
- Add any simple attributes of the M:N relationship to S

Step 6 Result:



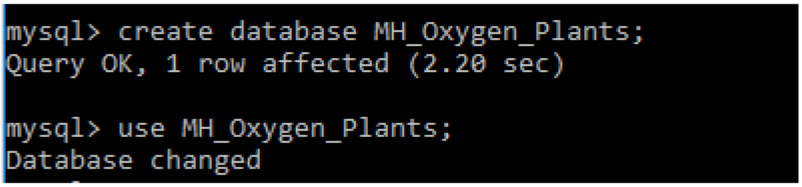
8. CREATING DATABASE USING MYSQL

An SQL schema is identified by a schema name, and includes an authorization identifier to indicate the user or account who owns the schema, as well as descriptors for each element in the schema. Schema elements include tables, constraints, views, domains, and other constructs (such as authorization grants) that describe the schema. A schema is created via the CREATE SCHEMA statement, which can include all the schema elements' definitions. Alternatively, the schema can be assigned a name and authorization identifier, and the elements can be defined later.

The CREATE TABLE command is used to specify a new relation by giving it a name and specifying its attributes and initial constraints. The attributes are specified first, and each attribute is given a name, a data type to specify its domain of values, and any attribute constraints, such as NOT NULL. The key, entity integrity, and referential integrity constraints can be specified within the CREATE TABLE statement after the attributes are declared, or they can be added later using the ALTER TABLE command.Following shows sample data definition statements in SQL for the MH_Oxygen_Plants relational database schema.

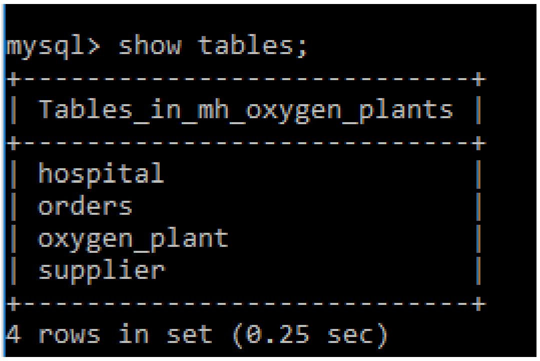
Screen Shots :

Create database :



```
mysql> create database MH_Oxygen_Plants;
Query OK, 1 row affected (2.20 sec)

mysql> use MH_Oxygen_Plants;
Database changed
```

Following are the tables in database :

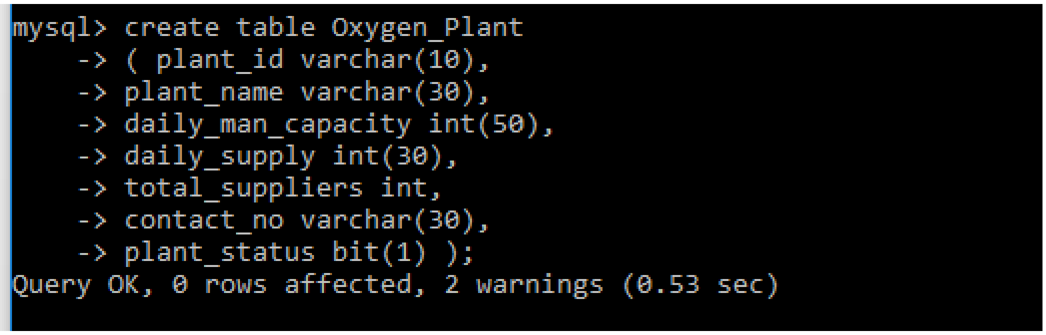


```
mysql> show tables;
+----------------------------+
| Tables_in_mh_oxygen_plants |
+----------------------------+
| hospital                   |
| orders                     |
| oxygen_plant               |
| supplier                   |
+----------------------------+
4 rows in set (0.25 sec)
```

9. Test Queries (Minimum 25 Queries)

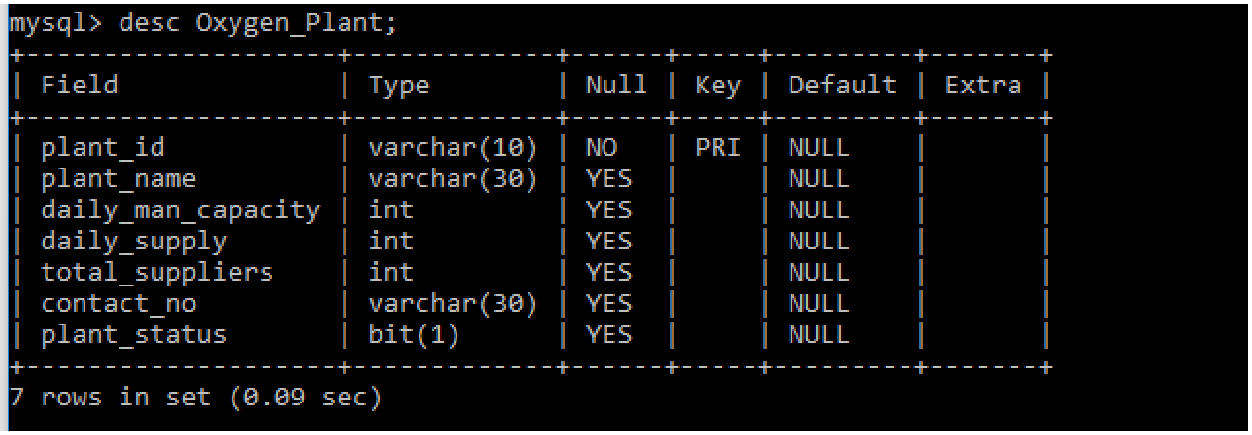
Screen Shots :

1. Create table Oxygen_Plant :



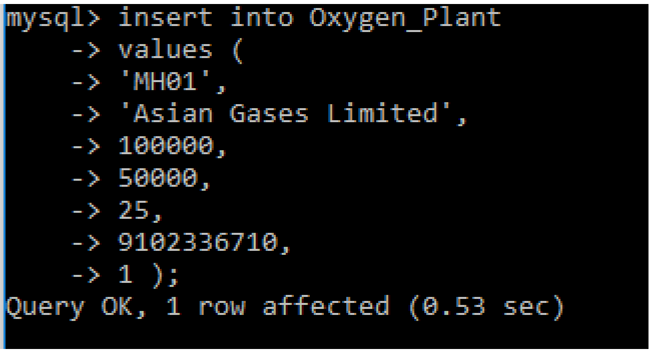
```
mysql> create table Oxygen_Plant
    -> ( plant_id varchar(10),
    -> plant_name varchar(30),
    -> daily_man_capacity int(50),
    -> daily_supply int(30),
    -> total_suppliers int,
    -> contact_no varchar(30),
    -> plant_status bit(1) );
Query OK, 0 rows affected, 2 warnings (0.53 sec)
```

2. Desc Oxygen_Plant Table :



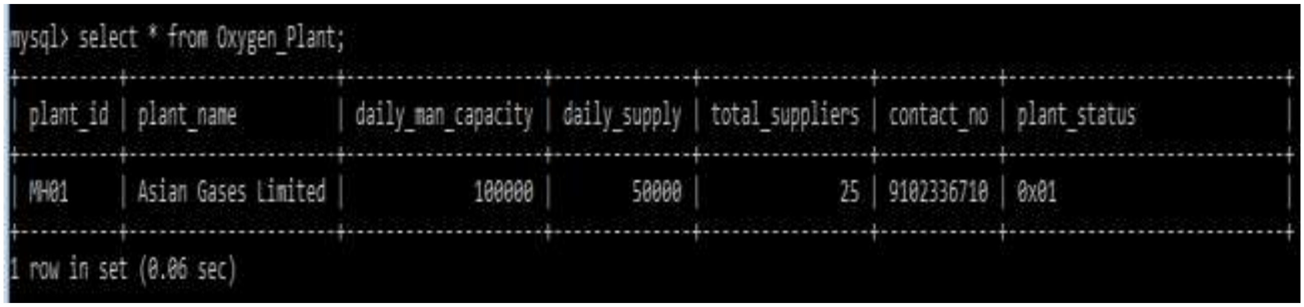
```
mysql> desc Oxygen_Plant;
+--------------------+-------------+------+-----+---------+-------+
| Field              | Type        | Null | Key | Default | Extra |
+--------------------+-------------+------+-----+---------+-------+
| plant_id           | varchar(10) | NO   | PRI | NULL    |       |
| plant_name         | varchar(30) | YES  |     | NULL    |       |
| daily_man_capacity | int         | YES  |     | NULL    |       |
| daily_supply       | int         | YES  |     | NULL    |       |
| total_suppliers    | int         | YES  |     | NULL    |       |
| contact_no         | varchar(30) | YES  |     | NULL    |       |
| plant_status       | bit(1)      | YES  |     | NULL    |       |
+--------------------+-------------+------+-----+---------+-------+
7 rows in set (0.09 sec)
```

3. Insert Records in table :



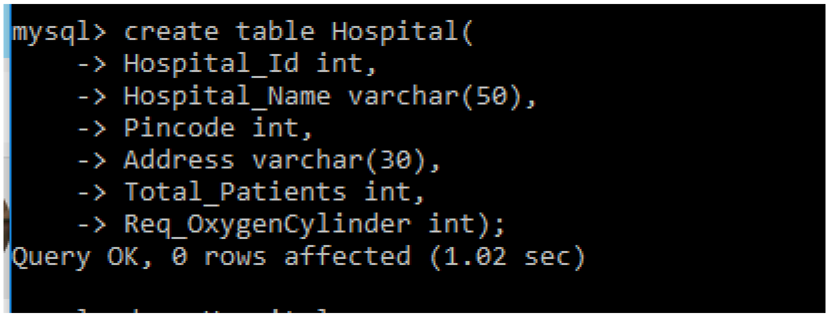
```
mysql> insert into Oxygen_Plant
    -> values (
    -> 'MH01',
    -> 'Asian Gases Limited',
    -> 100000,
    -> 50000,
    -> 25,
    -> 9102336710,
    -> 1 );
Query OK, 1 row affected (0.53 sec)
```

4. Display Records of Oxygen_Plant Table :



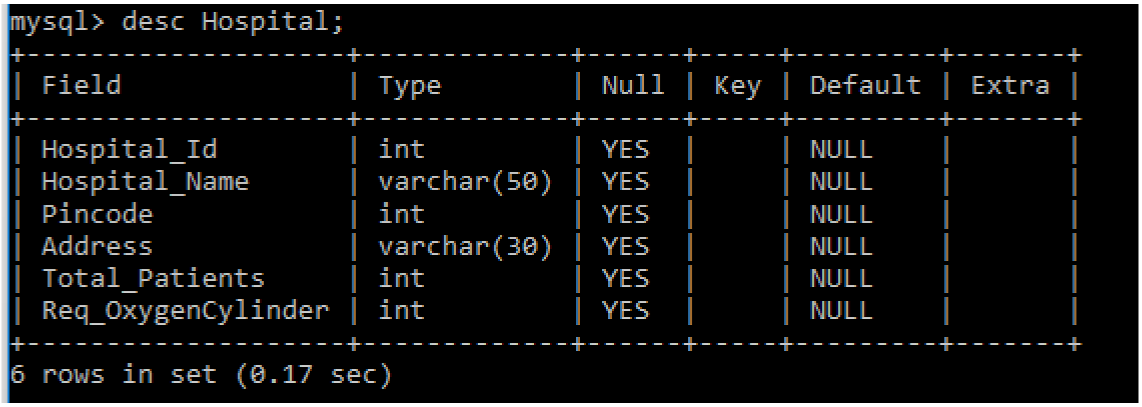
```
mysql> select * from Oxygen_Plant;
+----------+---------------------+--------------------+--------------+-----------------+------------+----------------------------+
| plant_id | plant_name          | daily_man_capacity | daily_supply | total_suppliers | contact_no | plant_status               |
+----------+---------------------+--------------------+--------------+-----------------+------------+----------------------------+
| MH01     | Asian Gases Limited |             100000 |        50000 |              25 | 9102336710 | 0x01                       |
+----------+---------------------+--------------------+--------------+-----------------+------------+----------------------------+
1 row in set (0.06 sec)
```

5. Create table Hospital:



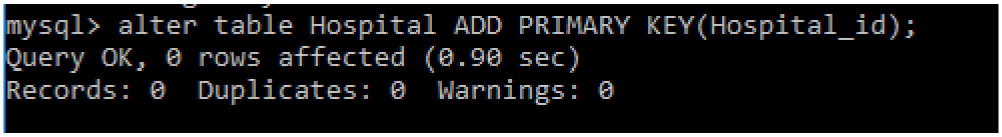
```
mysql> create table Hospital(
    -> Hospital_Id int,
    -> Hospital_Name varchar(50),
    -> Pincode int,
    -> Address varchar(30),
    -> Total_Patients int,
    -> Req_OxygenCylinder int);
Query OK, 0 rows affected (1.02 sec)
```

6. Describe Table :



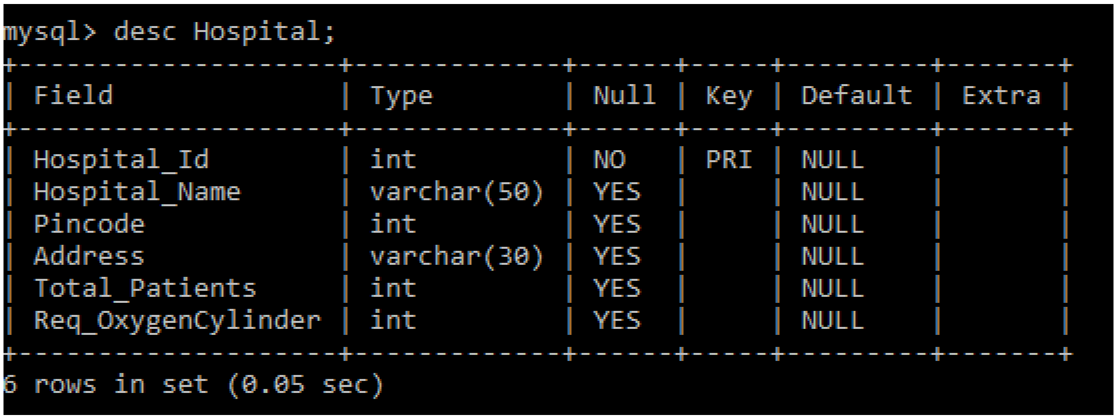
```
mysql> desc Hospital;
+--------------------+-------------+------+-----+---------+-------+
| Field              | Type        | Null | Key | Default | Extra |
+--------------------+-------------+------+-----+---------+-------+
| Hospital_Id        | int         | YES  |     | NULL    |       |
| Hospital_Name      | varchar(50) | YES  |     | NULL    |       |
| Pincode            | int         | YES  |     | NULL    |       |
| Address            | varchar(30) | YES  |     | NULL    |       |
| Total_Patients     | int         | YES  |     | NULL    |       |
| Req_OxygenCylinder | int         | YES  |     | NULL    |       |
+--------------------+-------------+------+-----+---------+-------+
6 rows in set (0.17 sec)
```

7. Add Primary key using Alter :



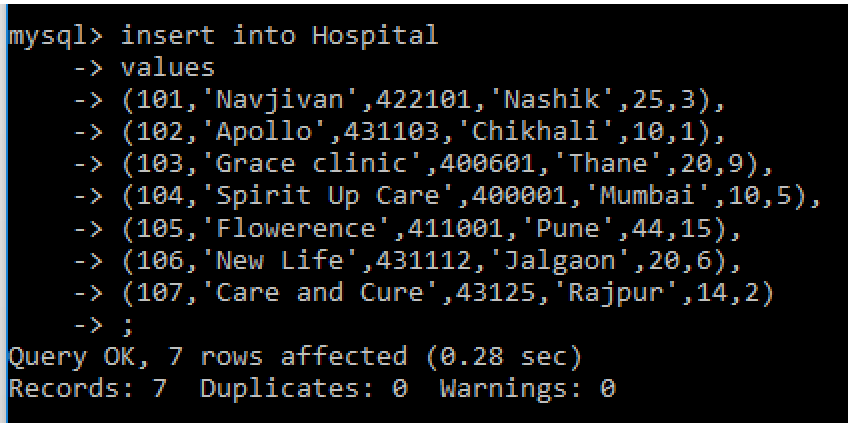
```
mysql> alter table Hospital ADD PRIMARY KEY(Hospital_id);
Query OK, 0 rows affected (0.90 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

8. Describe table :



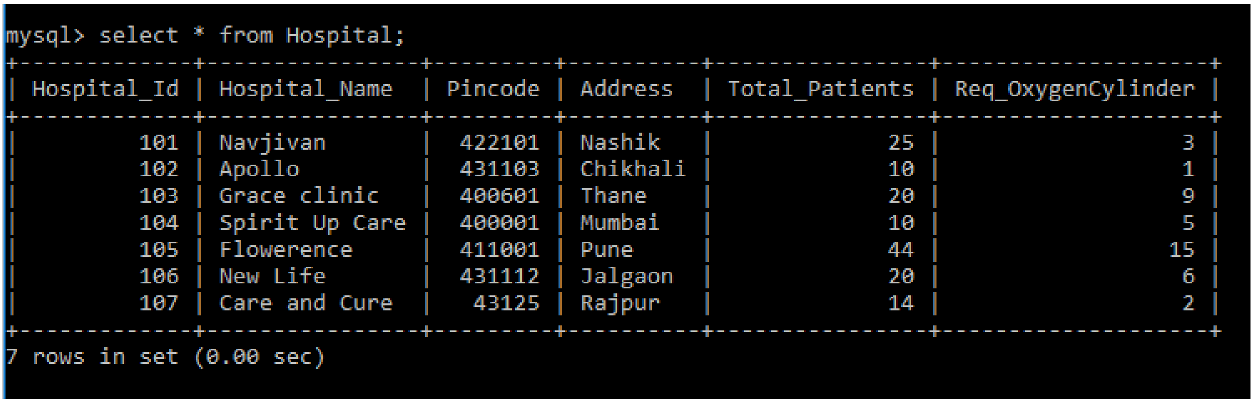
```
mysql> desc Hospital;
+--------------------+-------------+------+-----+---------+-------+
| Field              | Type        | Null | Key | Default | Extra |
+--------------------+-------------+------+-----+---------+-------+
| Hospital_Id        | int         | NO   | PRI | NULL    |       |
| Hospital_Name      | varchar(50) | YES  |     | NULL    |       |
| Pincode            | int         | YES  |     | NULL    |       |
| Address            | varchar(30) | YES  |     | NULL    |       |
| Total_Patients     | int         | YES  |     | NULL    |       |
| Req_OxygenCylinder | int         | YES  |     | NULL    |       |
+--------------------+-------------+------+-----+---------+-------+
6 rows in set (0.05 sec)
```

9. Insert Records :



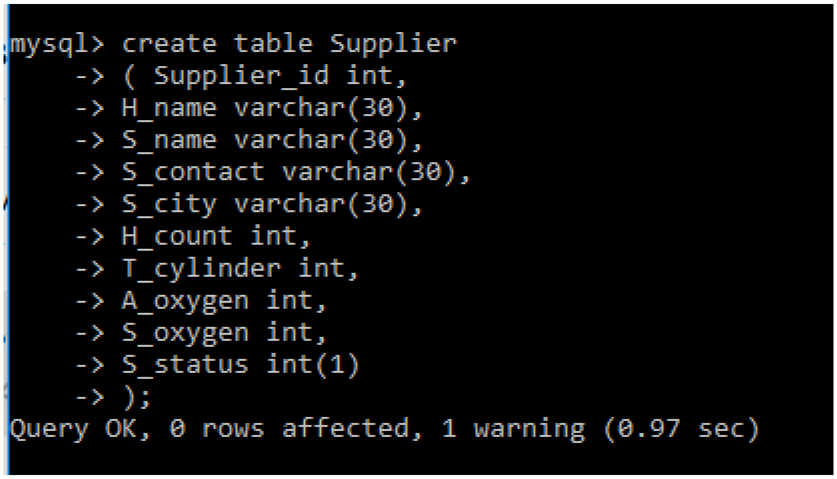
```
mysql> insert into Hospital
    -> values
    -> (101,'Navjivan',422101,'Nashik',25,3),
    -> (102,'Apollo',431103,'Chikhali',10,1),
    -> (103,'Grace clinic',400601,'Thane',20,9),
    -> (104,'Spirit Up Care',400001,'Mumbai',10,5),
    -> (105,'Flowerence',411001,'Pune',44,15),
    -> (106,'New Life',431112,'Jalgaon',20,6),
    -> (107,'Care and Cure',43125,'Rajpur',14,2)
    -> ;
Query OK, 7 rows affected (0.28 sec)
Records: 7  Duplicates: 0  Warnings: 0
```

10. Display Records :



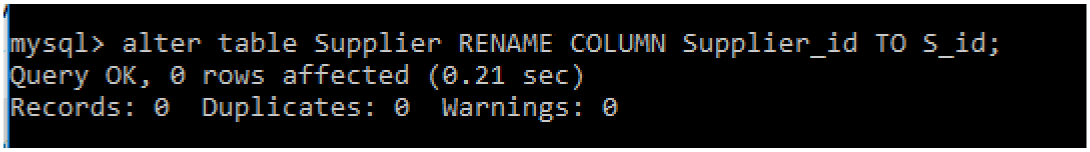
```
mysql> select * from Hospital;
+-------------+----------------+---------+----------+----------------+--------------------+
| Hospital_Id | Hospital_Name  | Pincode | Address  | Total_Patients | Req_OxygenCylinder |
+-------------+----------------+---------+----------+----------------+--------------------+
|         101 | Navjivan       |  422101 | Nashik   |             25 |                  3 |
|         102 | Apollo         |  431103 | Chikhali |             10 |                  1 |
|         103 | Grace clinic   |  400601 | Thane    |             20 |                  9 |
|         104 | Spirit Up Care |  400001 | Mumbai   |             10 |                  5 |
|         105 | Flowerence     |  411001 | Pune     |             44 |                 15 |
|         106 | New Life       |  431112 | Jalgaon  |             20 |                  6 |
|         107 | Care and Cure  |   43125 | Rajpur   |             14 |                  2 |
+-------------+----------------+---------+----------+----------------+--------------------+
7 rows in set (0.00 sec)
```

11. Create table Supplier :

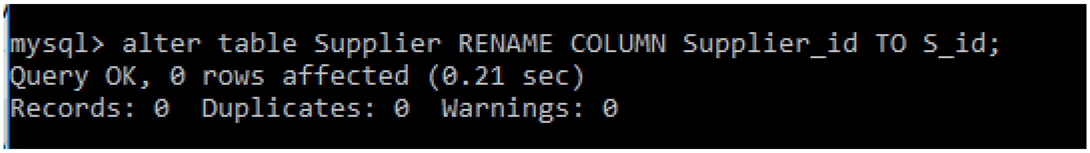


```
mysql> create table Supplier
    -> ( Supplier_id int,
    -> H_name varchar(30),
    -> S_name varchar(30),
    -> S_contact varchar(30),
    -> S_city varchar(30),
    -> H_count int,
    -> T_cylinder int,
    -> A_oxygen int,
    -> S_oxygen int,
    -> S_status int(1)
    -> );
Query OK, 0 rows affected, 1 warning (0.97 sec)
```

12. Insert Foreign keys using Alter :

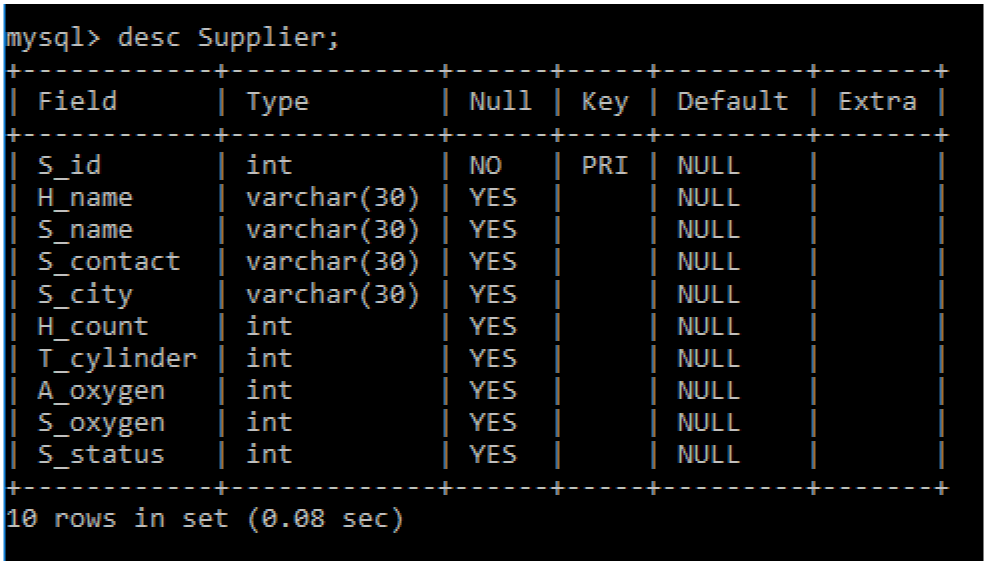


```
mysql> alter table Supplier RENAME COLUMN Supplier_id TO S_id;
Query OK, 0 rows affected (0.21 sec)
Records: 0  Duplicates: 0  Warnings: 0
```



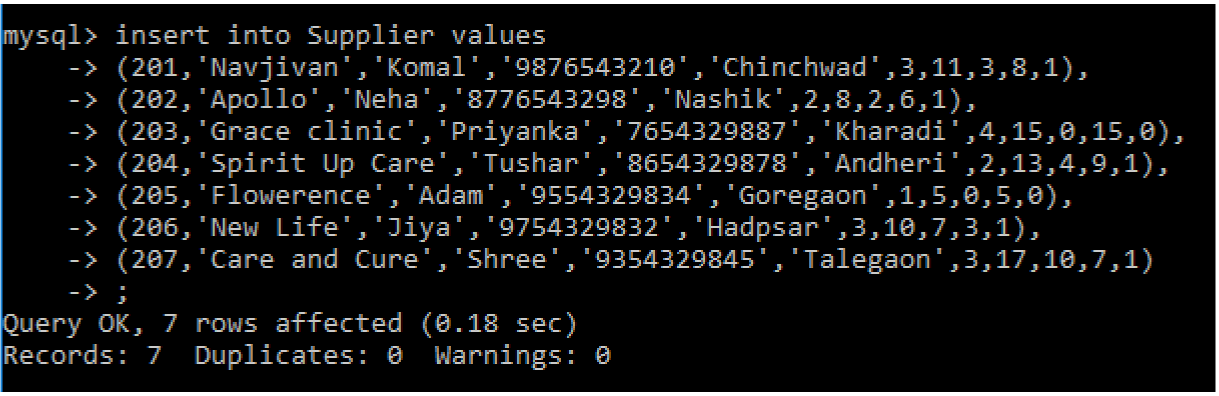
```
mysql> alter table Supplier RENAME COLUMN Supplier_id TO S_id;
Query OK, 0 rows affected (0.21 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

13. Describe table :



```
mysql> desc Supplier;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| S_id       | int         | NO   | PRI | NULL    |       |
| H_name     | varchar(30) | YES  |     | NULL    |       |
| S_name     | varchar(30) | YES  |     | NULL    |       |
| S_contact  | varchar(30) | YES  |     | NULL    |       |
| S_city     | varchar(30) | YES  |     | NULL    |       |
| H_count    | int         | YES  |     | NULL    |       |
| T_cylinder | int         | YES  |     | NULL    |       |
| A_oxygen   | int         | YES  |     | NULL    |       |
| S_oxygen   | int         | YES  |     | NULL    |       |
| S_status   | int         | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
10 rows in set (0.08 sec)
```

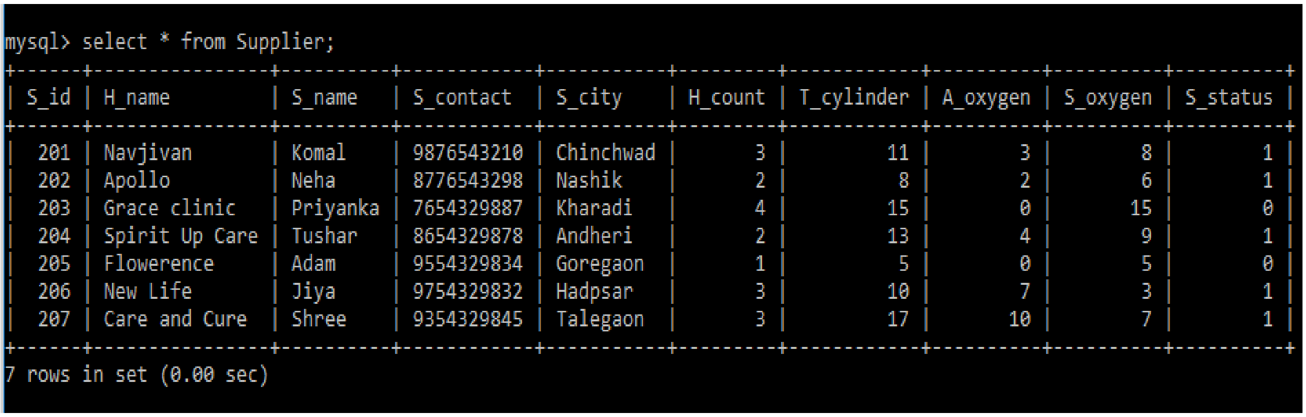
14. Insert Records :



```
mysql> insert into Supplier values
    -> (201,'Navjivan','Komal','9876543210','Chinchwad',3,11,3,8,1),
    -> (202,'Apollo','Neha','8776543298','Nashik',2,8,2,6,1),
    -> (203,'Grace clinic','Priyanka','7654329887','Kharadi',4,15,0,15,0),
    -> (204,'Spirit Up Care','Tushar','8654329878','Andheri',2,13,4,9,1),
    -> (205,'Flowerence','Adam','9554329834','Goregaon',1,5,0,5,0),
    -> (206,'New Life','Jiya','9754329832','Hadpsar',3,10,7,3,1),
    -> (207,'Care and Cure','Shree','9354329845','Talegaon',3,17,10,7,1)
    -> ;
Query OK, 7 rows affected (0.18 sec)
Records: 7  Duplicates: 0  Warnings: 0
```

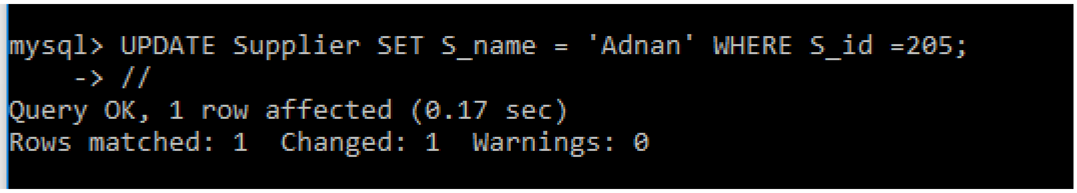
s

15. Display Records :

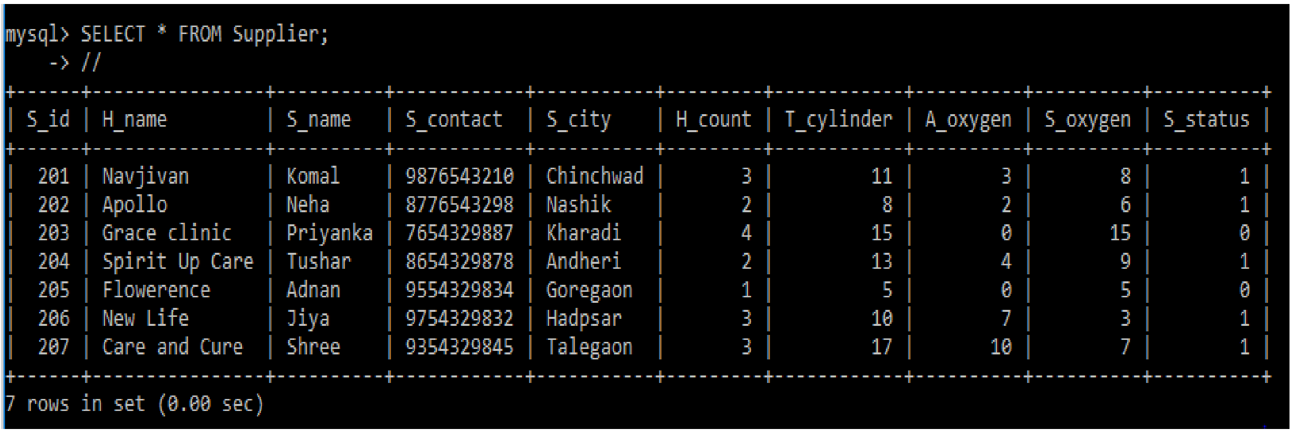


```
mysql> select * from Supplier;
+------+----------------+----------+------------+-----------+---------+------------+----------+----------+----------+
| S_id | H_name         | S_name   | S_contact  | S_city    | H_count | T_cylinder | A_oxygen | S_oxygen | S_status |
+------+----------------+----------+------------+-----------+---------+------------+----------+----------+----------+
|  201 | Navjivan       | Komal    | 9876543210 | Chinchwad |       3 |         11 |        3 |        8 |        1 |
|  202 | Apollo         | Neha     | 8776543298 | Nashik    |       2 |          8 |        2 |        6 |        1 |
|  203 | Grace clinic   | Priyanka | 7654329887 | Kharadi   |       4 |         15 |        0 |       15 |        0 |
|  204 | Spirit Up Care | Tushar   | 8654329878 | Andheri   |       2 |         13 |        4 |        9 |        1 |
|  205 | Flowerence     | Adam     | 9554329834 | Goregaon  |       1 |          5 |        0 |        5 |        0 |
|  206 | New Life       | Jiya     | 9754329832 | Hadpsar   |       3 |         10 |        7 |        3 |        1 |
|  207 | Care and Cure  | Shree    | 9354329845 | Talegaon  |       3 |         17 |       10 |        7 |        1 |
+------+----------------+----------+------------+-----------+---------+------------+----------+----------+----------+
7 rows in set (0.00 sec)
```

16. Update Supplier name :



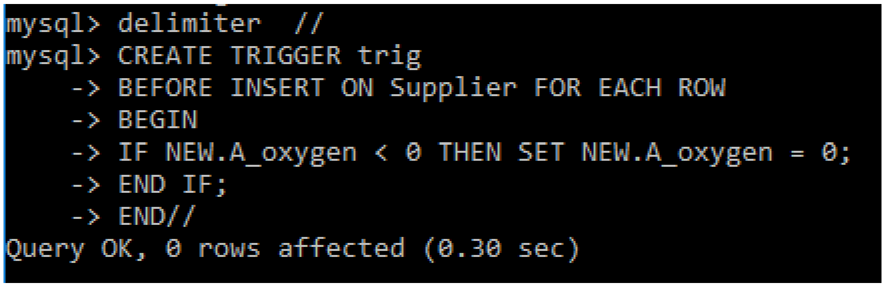
```
mysql> UPDATE Supplier SET S_name = 'Adnan' WHERE S_id =205;
    -> //
Query OK, 1 row affected (0.17 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```



```
mysql> SELECT * FROM Supplier;
    -> //
+------+----------------+----------+------------+-----------+---------+------------+----------+----------+----------+
| S_id | H_name         | S_name   | S_contact  | S_city    | H_count | T_cylinder | A_oxygen | S_oxygen | S_status |
+------+----------------+----------+------------+-----------+---------+------------+----------+----------+----------+
|  201 | Navjivan       | Komal    | 9876543210 | Chinchwad |       3 |         11 |        3 |        8 |        1 |
|  202 | Apollo         | Neha     | 8776543298 | Nashik    |       2 |          8 |        2 |        6 |        1 |
|  203 | Grace clinic   | Priyanka | 7654329887 | Kharadi   |       4 |         15 |        0 |       15 |        0 |
|  204 | Spirit Up Care | Tushar   | 8654329878 | Andheri   |       2 |         13 |        4 |        9 |        1 |
|  205 | Flowerence     | Adnan    | 9554329834 | Goregaon  |       1 |          5 |        0 |        5 |        0 |
|  206 | New Life       | Jiya     | 9754329832 | Hadpsar   |       3 |         10 |        7 |        3 |        1 |
|  207 | Care and Cure  | Shree    | 9354329845 | Talegaon  |       3 |         17 |       10 |        7 |        1 |
+------+----------------+----------+------------+-----------+---------+------------+----------+----------+----------+
7 rows in set (0.00 sec)
```

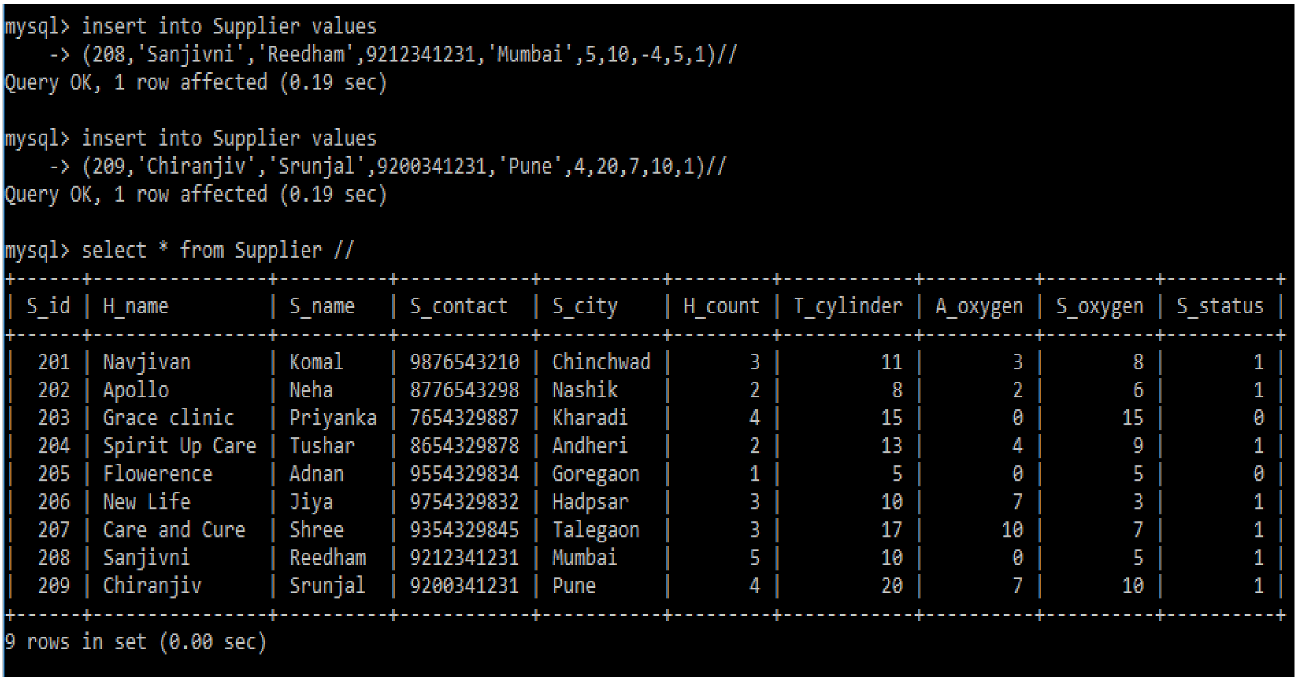
17. Create Trigger for inserting records and check supplied oxygen value should not less than 0. If you enter less than 0 value then it will assign 0 value :

Create Trigger :



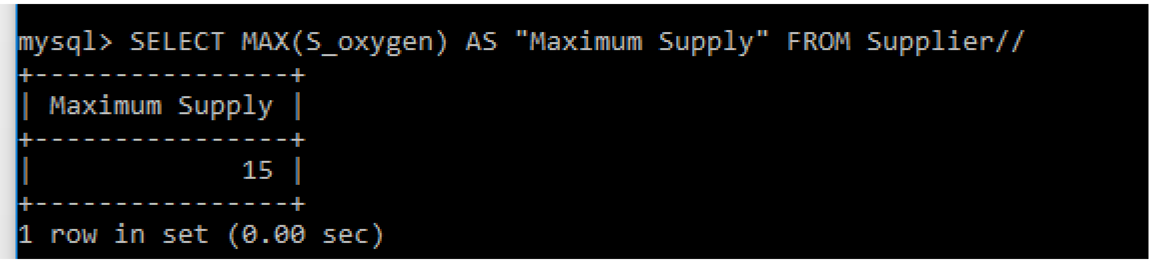
```
mysql> delimiter  //
mysql> CREATE TRIGGER trig
    -> BEFORE INSERT ON Supplier FOR EACH ROW
    -> BEGIN
    -> IF NEW.A_oxygen < 0 THEN SET NEW.A_oxygen = 0;
    -> END IF;
    -> END//
Query OK, 0 rows affected (0.30 sec)
```

Insert and Display Records :



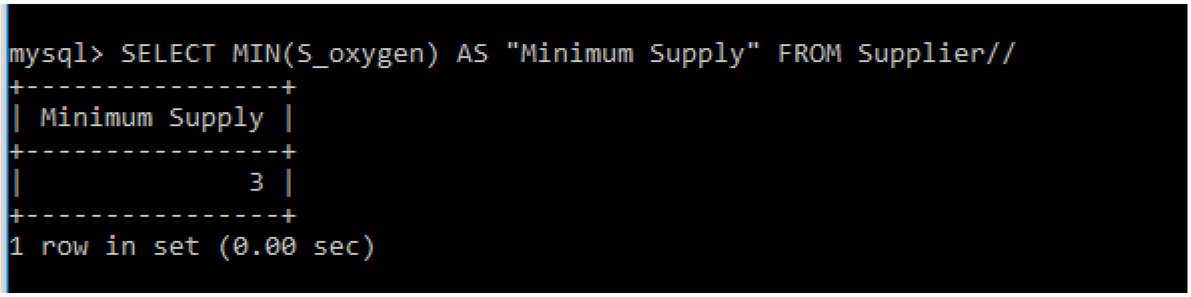
```
mysql> insert into Supplier values
    -> (208,'Sanjivni','Reedham',9212341231,'Mumbai',5,10,-4,5,1)//
Query OK, 1 row affected (0.19 sec)

mysql> insert into Supplier values
    -> (209,'Chiranjiv','Srunjal',9200341231,'Pune',4,20,7,10,1)//
Query OK, 1 row affected (0.19 sec)

mysql> select * from Supplier //
+------+--------------+----------+------------+-----------+---------+------------+----------+----------+----------+
| S_id | H_name       | S_name   | S_contact  | S_city    | H_count | T_cylinder | A_oxygen | S_oxygen | S_status |
+------+--------------+----------+------------+-----------+---------+------------+----------+----------+----------+
|  201 | Navjivan     | Komal    | 9876543210 | Chinchwad |       3 |         11 |        3 |        8 |        1 |
|  202 | Apollo       | Neha     | 8776543298 | Nashik    |       2 |          8 |        2 |        6 |        1 |
|  203 | Grace clinic | Priyanka | 7654329887 | Kharadi   |       4 |         15 |        0 |       15 |        0 |
|  204 | Spirit Up Care | Tushar | 8654329878 | Andheri   |       2 |         13 |        4 |        9 |        1 |
|  205 | Flowerence   | Adnan    | 9554329834 | Goregaon  |       1 |          5 |        0 |        5 |        0 |
|  206 | New Life     | Jiya     | 9754329832 | Hadpsar   |       3 |         10 |        7 |        3 |        1 |
|  207 | Care and Cure | Shree   | 9354329845 | Talegaon  |       3 |         17 |       10 |        7 |        1 |
|  208 | Sanjivni     | Reedham  | 9212341231 | Mumbai    |       5 |         10 |        0 |        5 |        1 |
|  209 | Chiranjiv    | Srunjal  | 9200341231 | Pune      |       4 |         20 |        7 |       10 |        1 |
+------+--------------+----------+------------+-----------+---------+------------+----------+----------+----------+
9 rows in set (0.00 sec)
```

18. Maximum number of supplied oxygen cylinder :



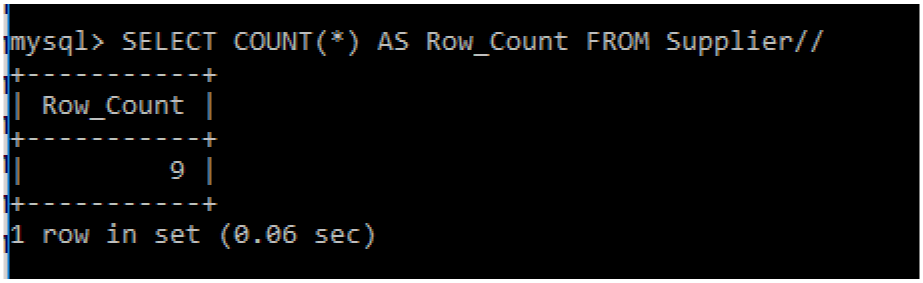
```
mysql> SELECT MAX(S_oxygen) AS "Maximum Supply" FROM Supplier//
+----------------+
| Maximum Supply |
+----------------+
|             15 |
+----------------+
1 row in set (0.00 sec)
```

19. Minimum number of supplied oxygen cylinder :



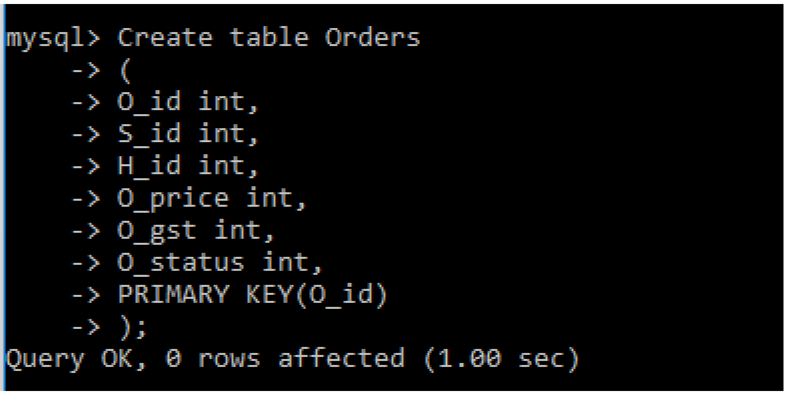
```
mysql> SELECT MIN(S_oxygen) AS "Minimum Supply" FROM Supplier//
+----------------+
| Minimum Supply |
+----------------+
|              3 |
+----------------+
1 row in set (0.00 sec)
```

20. Total number of suppliers in Asian Gases Limited :



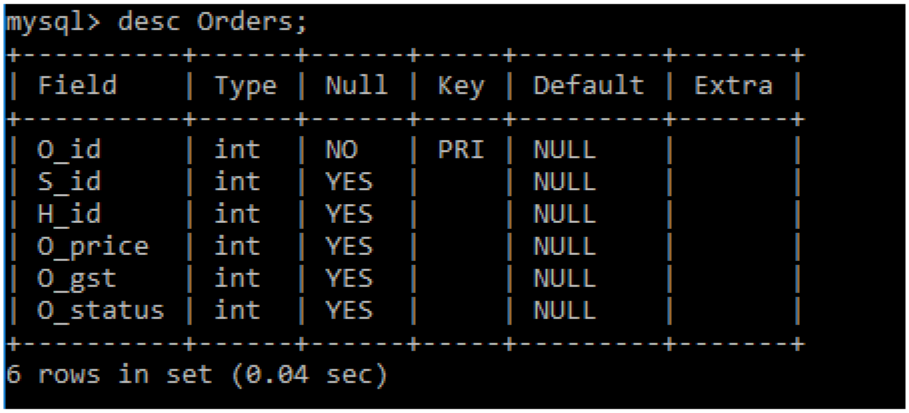
```
mysql> SELECT COUNT(*) AS Row_Count FROM Supplier//
+-----------+
| Row_Count |
+-----------+
|         9 |
+-----------+
1 row in set (0.06 sec)
```

21. Create table order :



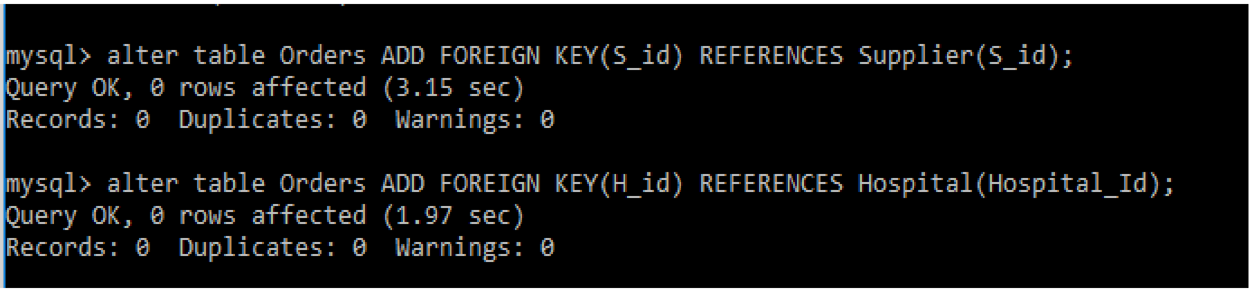
```
mysql> Create table Orders
    -> (
    -> O_id int,
    -> S_id int,
    -> H_id int,
    -> O_price int,
    -> O_gst int,
    -> O_status int,
    -> PRIMARY KEY(O_id)
    -> );
Query OK, 0 rows affected (1.00 sec)
```

22. Describe table :



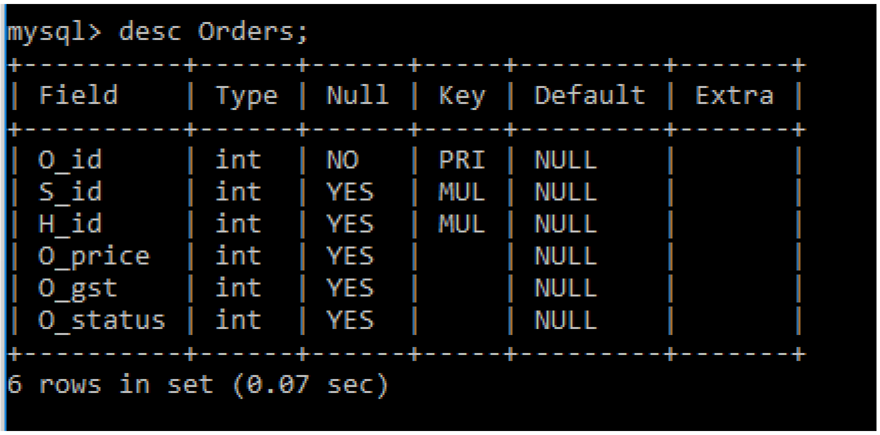
```
mysql> desc Orders;
+----------+------+------+-----+---------+-------+
| Field    | Type | Null | Key | Default | Extra |
+----------+------+------+-----+---------+-------+
| O_id     | int  | NO   | PRI | NULL    |       |
| S_id     | int  | YES  |     | NULL    |       |
| H_id     | int  | YES  |     | NULL    |       |
| O_price  | int  | YES  |     | NULL    |       |
| O_gst    | int  | YES  |     | NULL    |       |
| O_status | int  | YES  |     | NULL    |       |
+----------+------+------+-----+---------+-------+
6 rows in set (0.04 sec)
```

23. Add Foreign keys :



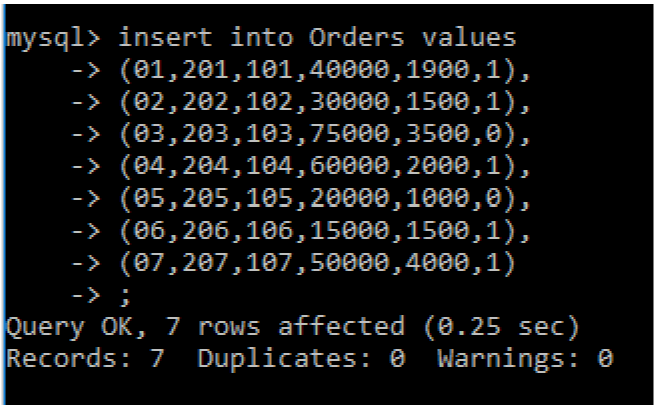
```
mysql> alter table Orders ADD FOREIGN KEY(S_id) REFERENCES Supplier(S_id);
Query OK, 0 rows affected (3.15 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> alter table Orders ADD FOREIGN KEY(H_id) REFERENCES Hospital(Hospital_Id);
Query OK, 0 rows affected (1.97 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

24. Describe table :



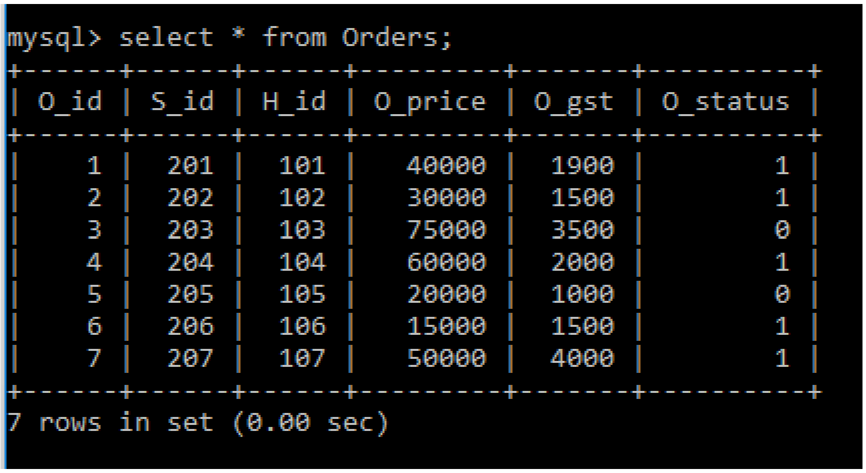
```
mysql> desc Orders;
+----------+------+------+-----+---------+-------+
| Field    | Type | Null | Key | Default | Extra |
+----------+------+------+-----+---------+-------+
| O_id     | int  | NO   | PRI | NULL    |       |
| S_id     | int  | YES  | MUL | NULL    |       |
| H_id     | int  | YES  | MUL | NULL    |       |
| O_price  | int  | YES  |     | NULL    |       |
| O_gst    | int  | YES  |     | NULL    |       |
| O_status | int  | YES  |     | NULL    |       |
+----------+------+------+-----+---------+-------+
6 rows in set (0.07 sec)
```

25. Insert Records :



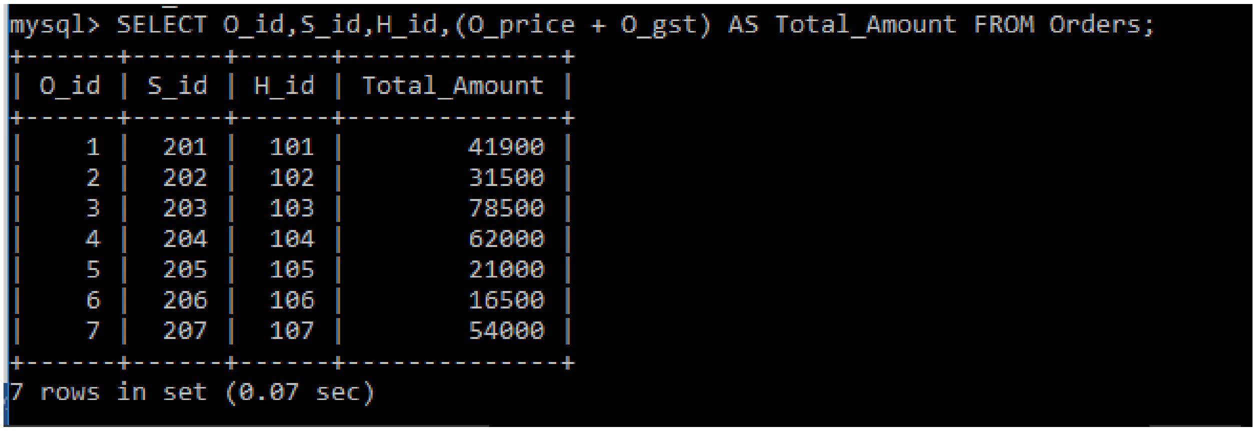
```
mysql> insert into Orders values
    -> (01,201,101,40000,1900,1),
    -> (02,202,102,30000,1500,1),
    -> (03,203,103,75000,3500,0),
    -> (04,204,104,60000,2000,1),
    -> (05,205,105,20000,1000,0),
    -> (06,206,106,15000,1500,1),
    -> (07,207,107,50000,4000,1)
    -> ;
Query OK, 7 rows affected (0.25 sec)
Records: 7  Duplicates: 0  Warnings: 0
```

26. Display Records :



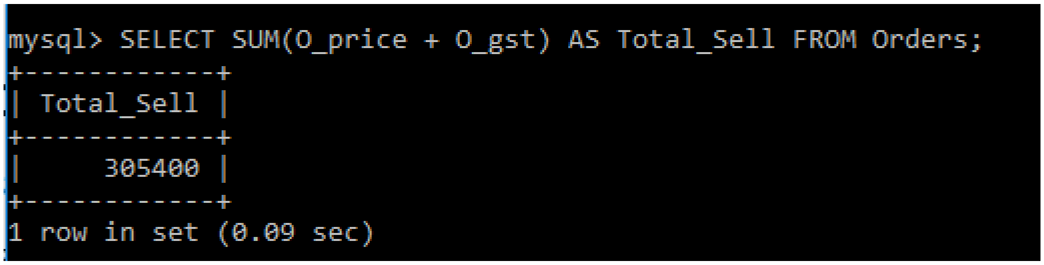
```
mysql> select * from Orders;
+------+------+------+---------+-------+----------+
| O_id | S_id | H_id | O_price | O_gst | O_status |
+------+------+------+---------+-------+----------+
|    1 |  201 |  101 |   40000 |  1900 |        1 |
|    2 |  202 |  102 |   30000 |  1500 |        1 |
|    3 |  203 |  103 |   75000 |  3500 |        0 |
|    4 |  204 |  104 |   60000 |  2000 |        1 |
|    5 |  205 |  105 |   20000 |  1000 |        0 |
|    6 |  206 |  106 |   15000 |  1500 |        1 |
|    7 |  207 |  107 |   50000 |  4000 |        1 |
+------+------+------+---------+-------+----------+
7 rows in set (0.00 sec)
```

27. Total amount of each order(price + gst) :



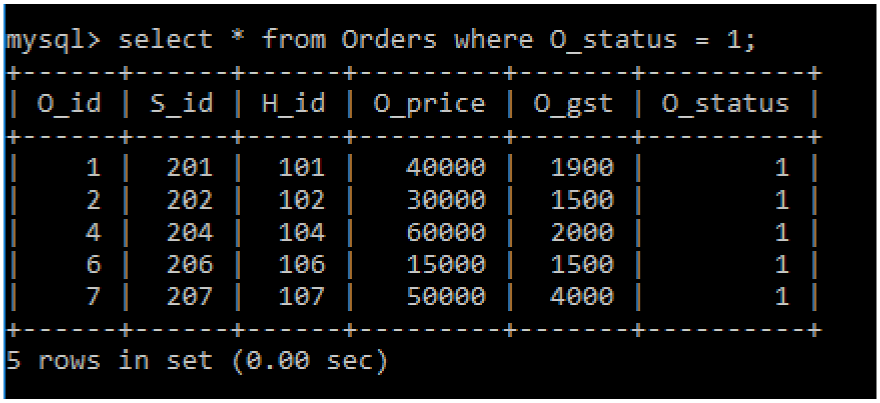
```
mysql> SELECT O_id,S_id,H_id,(O_price + O_gst) AS Total_Amount FROM Orders;
+------+------+------+--------------+
| O_id | S_id | H_id | Total_Amount |
+------+------+------+--------------+
|    1 |  201 |  101 |        41900 |
|    2 |  202 |  102 |        31500 |
|    3 |  203 |  103 |        78500 |
|    4 |  204 |  104 |        62000 |
|    5 |  205 |  105 |        21000 |
|    6 |  206 |  106 |        16500 |
|    7 |  207 |  107 |        54000 |
+------+------+------+--------------+
7 rows in set (0.07 sec)
```

28. Total sell of company :



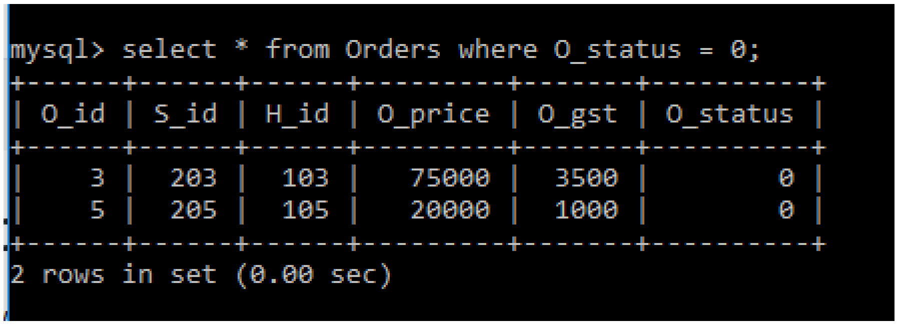
```
mysql> SELECT SUM(O_price + O_gst) AS Total_Sell FROM Orders;
+------------+
| Total_Sell |
+------------+
|     305400 |
+------------+
1 row in set (0.09 sec)
```

29. To get the information of orders which are dispatch :



```
mysql> select * from Orders where O_status = 1;
+------+------+------+---------+-------+----------+
| O_id | S_id | H_id | O_price | O_gst | O_status |
+------+------+------+---------+-------+----------+
|    1 |  201 |  101 |   40000 |  1900 |        1 |
|    2 |  202 |  102 |   30000 |  1500 |        1 |
|    4 |  204 |  104 |   60000 |  2000 |        1 |
|    6 |  206 |  106 |   15000 |  1500 |        1 |
|    7 |  207 |  107 |   50000 |  4000 |        1 |
+------+------+------+---------+-------+----------+
5 rows in set (0.00 sec)
```

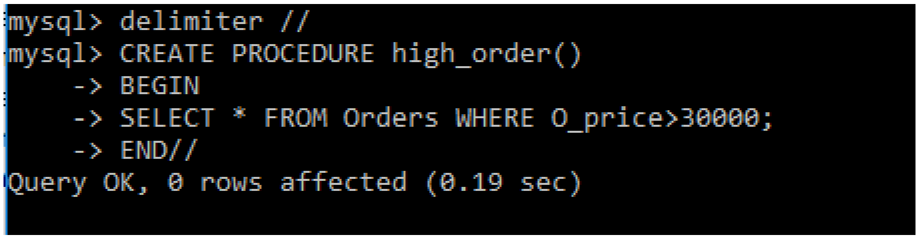
30. To get the information of orders which are not dispatch :



```
mysql> select * from Orders where O_status = 0;
+------+------+------+---------+-------+----------+
| O_id | S_id | H_id | O_price | O_gst | O_status |
+------+------+------+---------+-------+----------+
|    3 |  203 |  103 |   75000 |  3500 |        0 |
|    5 |  205 |  105 |   20000 |  1000 |        0 |
+------+------+------+---------+-------+----------+
2 rows in set (0.00 sec)
```

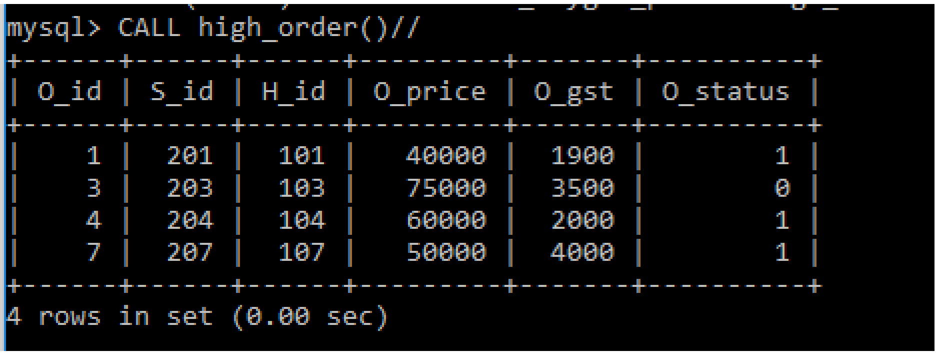
31. Procedure to get high amount of order from Orders table :

Create Procedure :



```
mysql> delimiter //
mysql> CREATE PROCEDURE high_order()
    -> BEGIN
    -> SELECT * FROM Orders WHERE O_price>30000;
    -> END//
Query OK, 0 rows affected (0.19 sec)
```

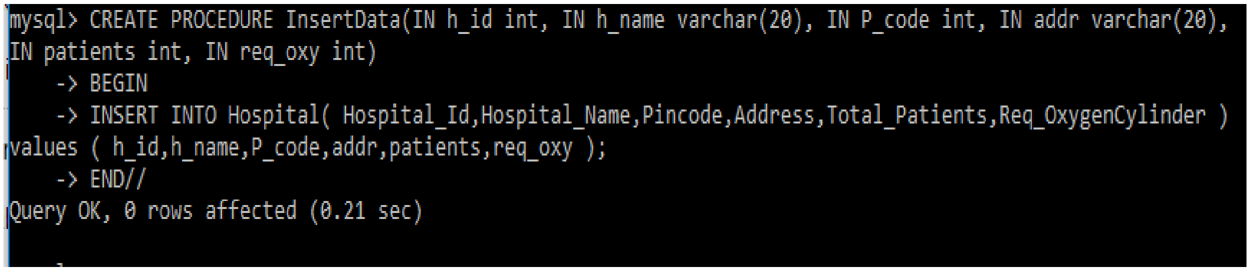
Call Procedure :



```
mysql> CALL high_order()//
+------+------+------+---------+-------+----------+
| O_id | S_id | H_id | O_price | O_gst | O_status |
+------+------+------+---------+-------+----------+
|    1 |  201 |  101 |   40000 |  1900 |        1 |
|    3 |  203 |  103 |   75000 |  3500 |        0 |
|    4 |  204 |  104 |   60000 |  2000 |        1 |
|    7 |  207 |  107 |   50000 |  4000 |        1 |
+------+------+------+---------+-------+----------+
4 rows in set (0.00 sec)
```

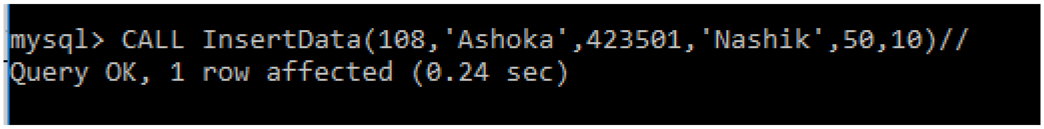
32. Procedure for insert new records in Hospital table :

Create Procedure :



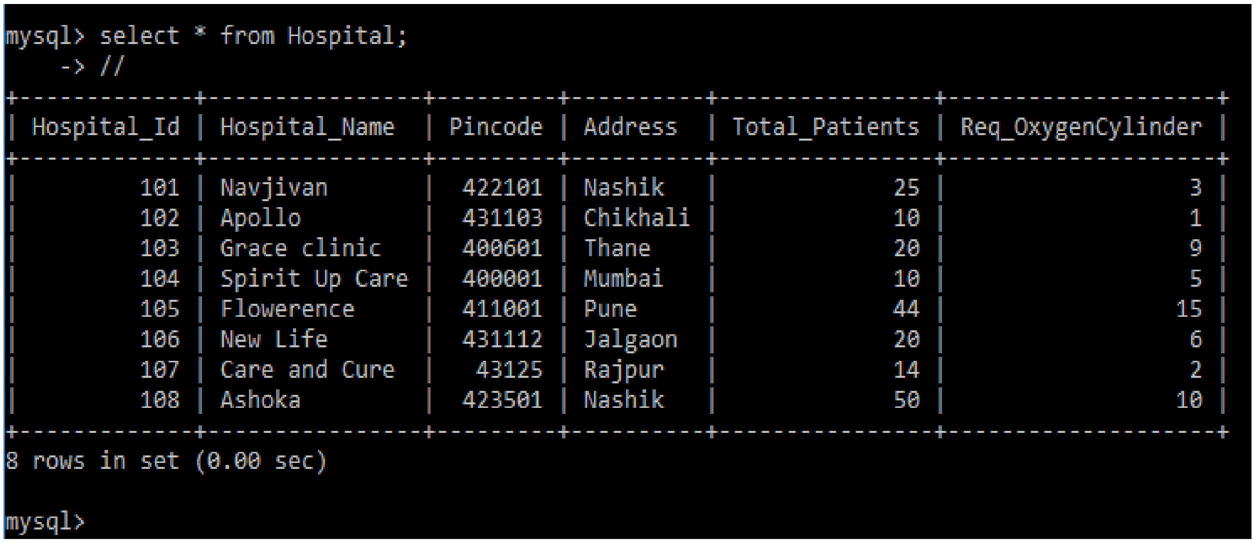
```
mysql> CREATE PROCEDURE InsertData(IN h_id int, IN h_name varchar(20), IN P_code int, IN addr varchar(20),
IN patients int, IN req_oxy int)
    -> BEGIN
    -> INSERT INTO Hospital( Hospital_Id,Hospital_Name,Pincode,Address,Total_Patients,Req_OxygenCylinder )
values ( h_id,h_name,P_code,addr,patients,req_oxy );
    -> END//
Query OK, 0 rows affected (0.21 sec)
```

Call Procedure :



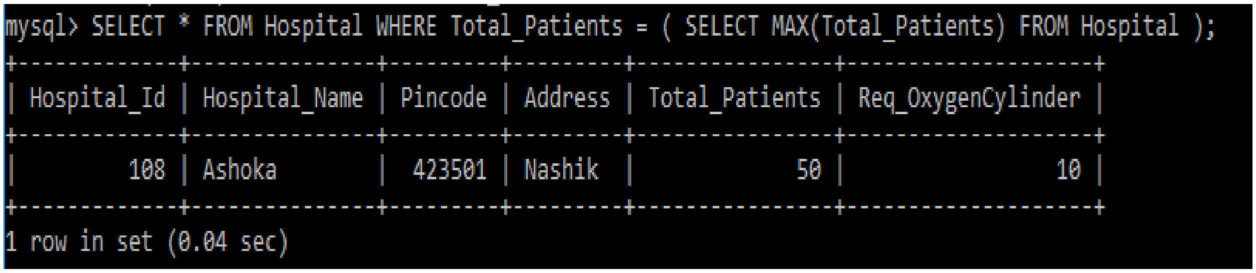
```
mysql> CALL InsertData(108,'Ashoka',423501,'Nashik',50,10)//
Query OK, 1 row affected (0.24 sec)
```

Display Records :



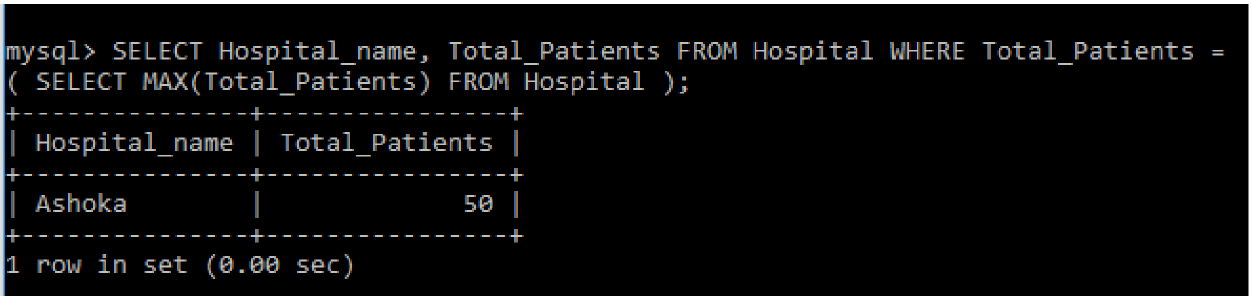
```
mysql> select * from Hospital;
    -> //
+-------------+-----------------+---------+----------+----------------+--------------------+
| Hospital_Id | Hospital_Name   | Pincode | Address  | Total_Patients | Req_OxygenCylinder |
+-------------+-----------------+---------+----------+----------------+--------------------+
|         101 | Navjivan        |  422101 | Nashik   |             25 |                  3 |
|         102 | Apollo          |  431103 | Chikhali |             10 |                  1 |
|         103 | Grace clinic    |  400601 | Thane    |             20 |                  9 |
|         104 | Spirit Up Care  |  400001 | Mumbai   |             10 |                  5 |
|         105 | Flowerence      |  411001 | Pune     |             44 |                 15 |
|         106 | New Life        |  431112 | Jalgaon  |             20 |                  6 |
|         107 | Care and Cure   |   43125 | Rajpur   |             14 |                  2 |
|         108 | Ashoka          |  423501 | Nashik   |             50 |                 10 |
+-------------+-----------------+---------+----------+----------------+--------------------+
8 rows in set (0.00 sec)

mysql>
```

33. To get information of hospital has maximum patients:



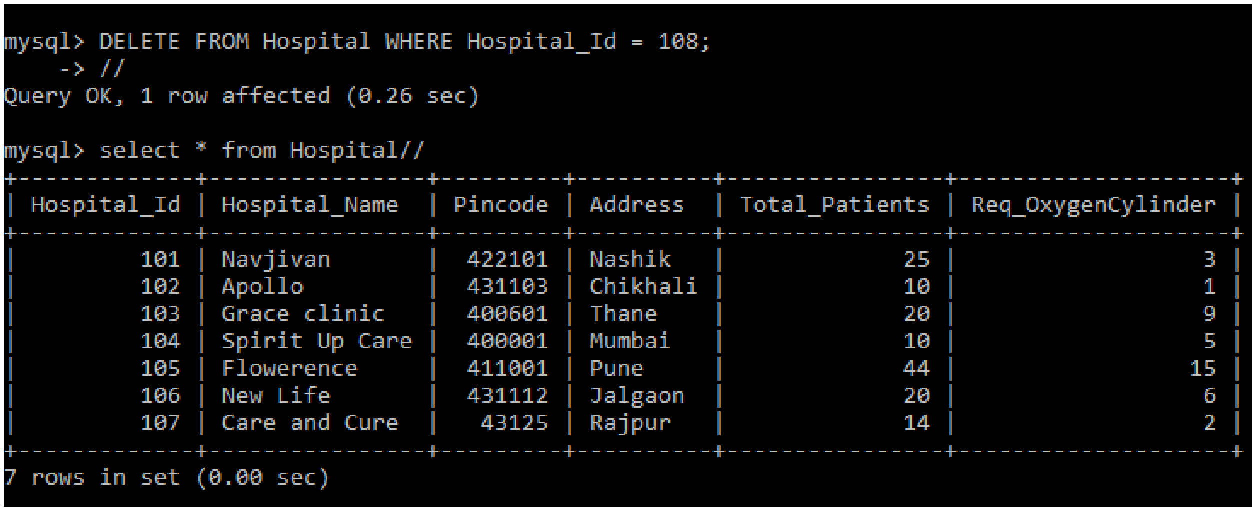
```
mysql> SELECT * FROM Hospital WHERE Total_Patients = ( SELECT MAX(Total_Patients) FROM Hospital );
+-------------+---------------+---------+---------+----------------+--------------------+
| Hospital_Id | Hospital_Name | Pincode | Address | Total_Patients | Req_OxygenCylinder |
+-------------+---------------+---------+---------+----------------+--------------------+
|         108 | Ashoka        |  423501 | Nashik  |             50 |                 10 |
+-------------+---------------+---------+---------+----------------+--------------------+
1 row in set (0.04 sec)
```

34. Name of Hospital has Maximum Patients :



```
mysql> SELECT Hospital_name, Total_Patients FROM Hospital WHERE Total_Patients =
( SELECT MAX(Total_Patients) FROM Hospital );
+---------------+----------------+
| Hospital_name | Total_Patients |
+---------------+----------------+
| Ashoka        |             50 |
+---------------+----------------+
1 row in set (0.00 sec)
```

35. Delete record from hospital :



```
mysql> DELETE FROM Hospital WHERE Hospital_Id = 108;
    -> //
Query OK, 1 row affected (0.26 sec)

mysql> select * from Hospital//
+-------------+----------------+---------+----------+----------------+--------------------+
| Hospital_Id | Hospital_Name  | Pincode | Address  | Total_Patients | Req_OxygenCylinder |
+-------------+----------------+---------+----------+----------------+--------------------+
|         101 | Navjivan       |  422101 | Nashik   |             25 |                  3 |
|         102 | Apollo         |  431103 | Chikhali |             10 |                  1 |
|         103 | Grace clinic   |  400601 | Thane    |             20 |                  9 |
|         104 | Spirit Up Care |  400001 | Mumbai   |             10 |                  5 |
|         105 | Flowerence     |  411001 | Pune     |             44 |                 15 |
|         106 | New Life       |  431112 | Jalgaon  |             20 |                  6 |
|         107 | Care and Cure  |   43125 | Rajpur   |             14 |                  2 |
+-------------+----------------+---------+----------+----------------+--------------------+
7 rows in set (0.00 sec)
```

10. Conclusion

In this project we develop a system to manage oxygen cylinder supply in Maharashtra. Several individuals and elderly are intending and forcing oxygrn suppliers to provide oxygen cylinder for residence use. While this is done to avoid the rush for oxygen in case an emergency arises, particularly in COVID situation. This project gives data about in which hospital oxygen cylinders are available.

We use trigger for inserting the data of supplier and it validate that supplier should have at least supply one oxygen cylinder. We use procedure to get highest amount of order and insert the entry of new hospital in hospital table. We perform all DDL and DML queries to perform operation on database.

From this project we can manage the supply of oxygen cylinder, get the records of suppliers and hospitals. During our database management course we have learned about the basics of database design. This project gave us the opportunity to try our new skills in practice. While doing this project we also gained deeper understanding on database design and how it can be implemented in real life situations.

11. References

- tutorialspoint.com
- en.wikipedia.org/wiki/Database_management_system
- https://www.w3schools.com/mysql/
- Database System concepts(Abraham Silberschatz) – Book
- https://en.wikipedia.org/wiki/MySQL
- https://creately.com/
- Dr. P. S. Deshpande, "SQL and PL/SQL for oracle 10g Black Book" , DreamTech