GROUP - 9
DAMG6210.18472.202210
Project Submission 5
Database Management and Database Design

PROJECT TOPIC: **OXYGEN RENTAL SYSTEM**

PROJECT MEMBERS:

S.No	Name	NUID	Email Address
1.	Namratha Gonuguntla	001565081	Gonuguntla.n@northeastern.edu
2.	Nitin Dornipadu	002197358	Dornipadu.n@northeastern.edu
3.	Rajesh Kaireddy	002198556	Kaireddy.r@northeastern.edu
4.	Sai Kumar Ganga	002191288	Ganga.s@northeastern.edu

PROBLEM STATEMENT:

As 2nd wave of covid-19 is ravaging the lives of people in USA, the occurrence of clinical oxygen shortage acted as catalyst to the deaths across Massachusetts counties. The purpose of the database is to store, maintain and process data amidst the pandemic. The main aim is to provide a platform where people can connect to the nearest oxygen suppliers for oxygen cylinders in emergency.

OBJECTIVES:

- Maintain records of real-time oxygen availability status across multiple oxygen industry plants.
- Maintain (Insert, Update, delete) data of Oxygen plant locations and Customers.
- Maintain the Booking History (Unique transaction ID, Customer ID, plant ID, quantity) to visualise and generate a report on demand of oxygen across county.
- Security: Providing them Authorization based on role for security purposes.
    - Below are the possible authorization roles:
        1. Customer role/User
        2. Oxygen supplier role
        3. Admin
- Referential integrity: When the customer places an order, his unique-ID, oxygen plant Id are stored as foreign keys in a single table called Order. In addition, foreign keys will be used wherever needed to maintain the integrity of the data.

- Data Redundancy: Trying to reduce data redundancy by providing Unique IDs for customer, plant, cylinder and orders.

PROBLEM SOLUTION:

The Covid-19 has taken a striking toll on people all over the world, affecting health physically and mentally. Short of breath being one of the major symptom impacting the lungs causing acute respiratory distress and failure. Due to this, demand for Oxygen has increased a great deal and difficulty to access an oxygen cylinder escalated. The Oxygen rental system project is to connect customers (patients) to suppliers based on location to make access to Oxygen easier.

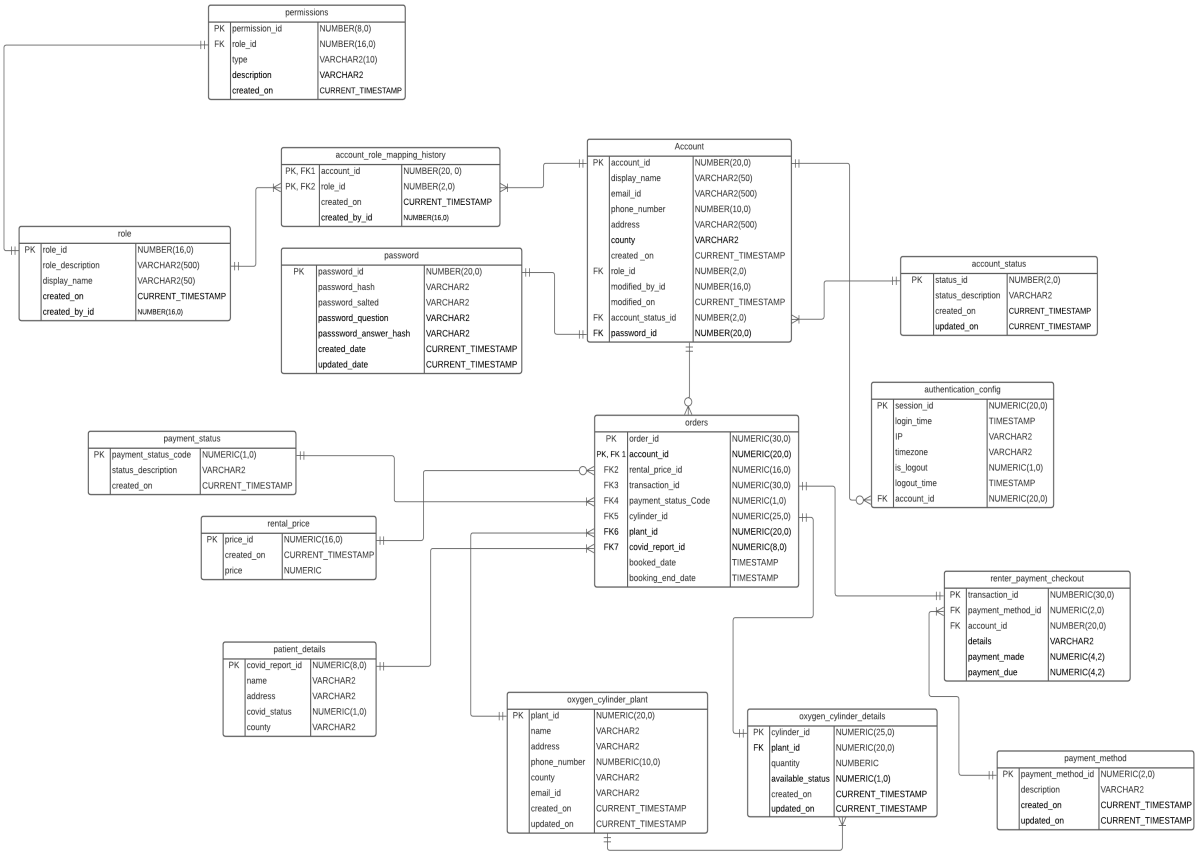
DATABASE DESIGN DOCUMENT

The main purpose of the design document is to list out the requirements of Oxygen rental System project. It helps us to gather and analyse the ideas gathered. The document is subject to change, if the requirement add up to the project. Currently it is mainly prepared to set foundation for the design phase. This document prepared is the first version of objectives document and helps us convert the logical/ conceptual model to physical model (tables) for the project.

Design Requirement

- Specify the primary key in each table by specifying PK beside the attribute(s) fields.
- Specify the foreign key in each table by specifying FK beside the attribute(s) fields.
- Draw a line between the fields of each table to represent relationships between the tables and entities. The line drawn must be pointed directly to the fields in each table that determine the relationship.
- Apply Crow's Foot Notation
    - Entities are represented as boxes, and relationships are represented as lines between them.
    - The symbols used define the cardinality of the relation. Symbols used are ring (zero), dash (one), crow's foot (many).
- Specify which table is comes on which side of the relationship. (if one, place a dash, if many side place a crow's feet symbol next to the field where. The line ends)

ENTITY RELATIONSHIP DIAGRAM (ER-diagram)



DOCUMENT ENTITY AND ATTRIBUTES WITH DATA TYPES

Account

Attribute	Datatype	Comments	Description
account_id	NUMBER(20,0)	Primary Key	Unique ID for account
display_name	VARCHAR2(50)	Unique, Not Null	Name displayed for the account
email_id	VARCHAR2(500)		Email Id of the user
phone_number	NUMBER(10,0)		Contact number of the user
address	VARCHAR2(500)		Address of the user
county	VARCHAR2		County of the user
created_on	CURRENT_TIMESTAMP		Date when account was created
role_id	NUMBER(2,0)	Foreign Key	Role ID of account
modified_by_id	NUMBER(16,0)		Modified by what id
modified_on	CURRENT_TIMESTAMP		Time when something was modified
Account_status_id	Number(2,0)	Foreign Key	Status of account
password_id	NUMBER(20,0)	Foreign Key	Password ID of the user

Account_Status

Attribute	Datatype	Comments	Description
account_status_id	NUMBER(2,0)	Primary Key	Unique ID for status of an account
status_description	VARCHAR2		Description of the account status
updated_on	CURRENT_TIMESTAMP		Time when account was updated
created_on	CURRENT_TIMESTAMP		Time when account was created

Authentication_config

Attribute	Datatype	Comments	Description
session_id	NUMBER(20,0)	Primary Key	Unique ID for session of an account
login_time	TIMESTAMP		Time when account is logged in
ip	VARCHAR2		IP address
timezone	VARCHAR2		TimeZone where account logged in
is_logout	NUMERIC(1,0)		Logout status
logout_time	TIMESTAMP		Time when account is logged out
account_id	NUMERIC(20,0)	Foreign Key	Unique Account ID of the account.

Orders

Attribute	Datatype	Comments	Description
order_id	NUMERIC(30,0)	Primary Key	Unique Order ID of purchase
account_id	NUMERIC(20,0)	Foreign Key	Unique Account ID of user
cylinder_id	NUMERIC(25,0)	Foreign Key	Unique Cylinder ID of order
rental_price_id	NUMERIC(16,0)	Foreign Key	Price ID of the rental
transaction_id	NUMERIC(30,0)	Foreign Key	Transaction id of order
payment_status_code	NUMERIC(1,0)	Foreign Key	Payment status of the order
plant_id	NUMERIC(20,0)	Foreign Key	Unique Plant ID of the cylinder
covid_report_id	NUMERIC(8,0)	Foreign Key	Covid report id of the patient
booked_date	TIMESTAMP		Booking date of the order
booked_end_date	TIMESTAMP		Booking end date on the order

Permissions

Attribute	Datatype	Comments	Description
permission_id	NUMBER(8,0)	Primary Key	Permission ID of the account
role_id	NUMBER(16,0)	Foreign Key	Role id of the user
type	VARCHAR2(10)		Type of permissions
description	VARCHAR2		Description of permissions
created_on	CURRENT_TIMESTAMP		Time when account is created

Oxygen_Cylinder_Details

Attribute	Datatype	Comments	Description
cylinder_id	NUMERIC(25,0)	Primary Key	Unique ID of the cylinder
Plant_id	NUMERIC(20,0)	Foreign Key	Unique ID of the cylinder plant
quantity	NUMERIC		Quantity of order
created_on	CURRENT_TIMESTAMP		Time when details are created
updated_on	CCURRENT_TIMESTAMP		Time when details are updated
available_status	NUMERIC(1,0)		Availability Status of cylinder

Oxygen_Cylinder_Plant

Attribute	Datatype	Comments	Description
plant_id	NUMERIC(20,0)	Primary Key	Unique ID of the cylinder plant
name	VARCHAR2		Name of the plant
address	VARCHAR2		Address of the plant
phone_number	NUMERIC(10,0)		Contact number of the plant
state	VARCHAR2		State where plant is located
email_id	VARCHAR2		Email id of the plant
created_on	TIMESTAMP		Time when created
updated_on	TIMESTAMP		Time when updated

Renter_Payment_Checkout

Attribute	Datatype	Comments	Description
transaction_id	NUMERIC(30,0)	Primary Key	Unique ID of the transaction
payment_method_id	NUMERIC(2,0)	Foreign Key	Payment ID of order
account_id	NUMBER(20,0)	Foreign Key	Unique ID of the account
details	VARCHAR2		Details of Payment
payment_made	NUMERIC(4,2)		Time when payment was made
payment_due	NUMERIC(4,2)		Time by when payment must be made

Payment_Method

Attribute	Datatype	Comments	Description
payment_method_id	NUMERIC(2,0)	Primary Key	Payment ID of order
description	VARCHAR2		Description of payment method used
created_on	CURRENT_TIMESTAMP		Time when created
updated_on	CURRENT_TIMESTAMP		Time when updated

Role

Attribute	Datatype	Comments	Description
role_id	NUMBER(16,0)	Primary Key	Unique ID for role of an account
role_description	VARCHAR2(500)		Description of the account status
display_name	VARCHAR2(50)		Name displayed for the account
created_on	CURRENT_TIME_STAMP		Time when role was created
created_by_id	NUMBER(16,0)		Person id was created

Account _Role _Mapping

Attribute	Datatype	Comments	Description
account_id	NUMBER(20,0)	Primary Key	Unique ID for account role of an account
role_id	NUMBER(2,0)	Primary Key	Unique ID for role of an account
Created_on	CURRENT_TIMESTAMP		Time when role was created
created_by_id	NUMBER(16,0)		Person id was created

Password

Attribute	Datatype	Comments	Description
password_id	NUMBER(16,0)	Primary Key	Unique ID for password of an account
password_hash	VARCHAR2(50)		Password hashing add a layer to protect login credentials.
password_salted	VARCHAR2(50)		password salted is created to protect the passwords by adding an extra bits of data
password_question	VARCHAR2(50)		password question is asked for two factor authentication.
password_answer_hash	VARCHAR2(50)		password answer maintains the integrity for the intended users.
created_date	CURRENT_TIMESTAMP		Date when password was created
Updated_date	CURRENT_TIMESTAMP		Date when password was updated

Payment_Status

Attribute	Datatype	Comments	Description
payment_status_code	NUMERIC(1,0)	Primary Key	Status code of payment
status_description	VARCHAR2		Description of payment status
created_on	CURRENT_TIMESTAMP		Time when created

Rental_Price

Attribute	Datatype	Comments	Description
price_id	NUMERIC(16,0)	Primary Key	Price ID of Order
created_on	CURRENT_TIMESTAMP		Time when created
price	NUMERIC		Price value details

Patient_Details

Attribute	Datatype	comments	Description
covid_report_id	NUMERIC(8,0)	Primary Key	Covid report id of the patient
name	VARCHAR2		Name of patient
address	VARCHAR2		Address of patient
covid_status	NUMERIC(1,0)		Covid result status of patient
county	VARCHAR2		County details of Patient

BUSINESS RULES:

- Each **USER** can be associated with only one **ACCOUNT**, each **ACCOUNT** is associated with one/ more **ROLES** and Each **ROLE** is associated with only one **PERMISSION_ID**.
- **USER** can take only one **ROLE** at any point of time.
- When an **ACCOUNT'S ROLE** has been updated, the update is stored as new row in **ACCOUNT_ROLE_MAPPING_HISTORY**.
- Status of the account is tracked in **ACCOUNT_STATUS** table with created datetime and updated datetime.
- All the **USER PASSWORDS** are salted and hashed(encrypted) before storing the data.
- Each **USER** should have a valid covid test **CERTIFICATE_ID** to put an order.
- In an order, only one **CYCLINDER** can be placed at a time irrespective of the plant.
- Payment checkout method options are available, each order can have one **PAYMENT CHECKOUT**. Payment status table gives us the status details of payment.

SECURITY:

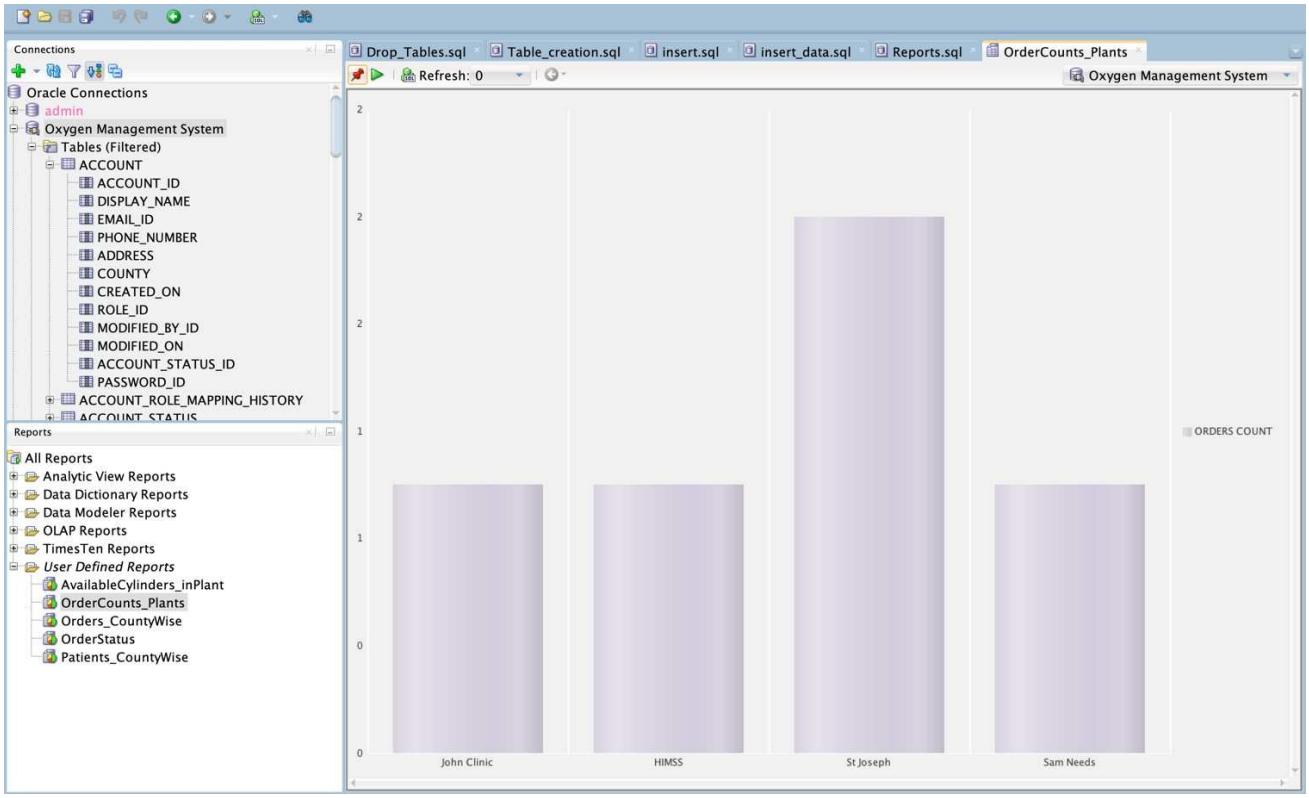
- Database Security means protecting the data from threats and unusual activity from unauthorized parties. In our project we are providing access level security to ensure right permissions and access to keep business data safe.
- Each account **(account_id)** is mapped to a role **(role_id)** based on the user requirement as a customer or supplier.
- Each role is given a set of permissions **(permission_id)** based on the role_id to authorize.
- Password will be in encrypted form. We are implementing this by using password encryption algorithm (DES_CBC_PKCS5)
- We are also providing a login-logout option to maintain integrity along with security based on Authorization roles. Each account (**account_id**) has a **unique session_id**, it can have zero or multiple sessions based on the login/ logout authentication.

- Below are the authorization roles:
    1. **Customer role/User** :
        - A customer places an order for an oxygen cylinder based on the requirement and availability.
        - Has Read access to Cylinder and Plant details.
        - Has Read/ Write/ Update Access to Orders and Payment.
    2. **Oxygen supplier role** :
        - A supplier is responsible for maintaining the cylinder and plant details as per location.
        - Has Read access to Orders.
        - Has Read/ Write/ Update Access to Cylinder and Plant details.
    3. **Admin :**
        - An admin is responsible for maintain the customer and supplier accounts and orders.
        - Has All level of Access to all entities in the database.

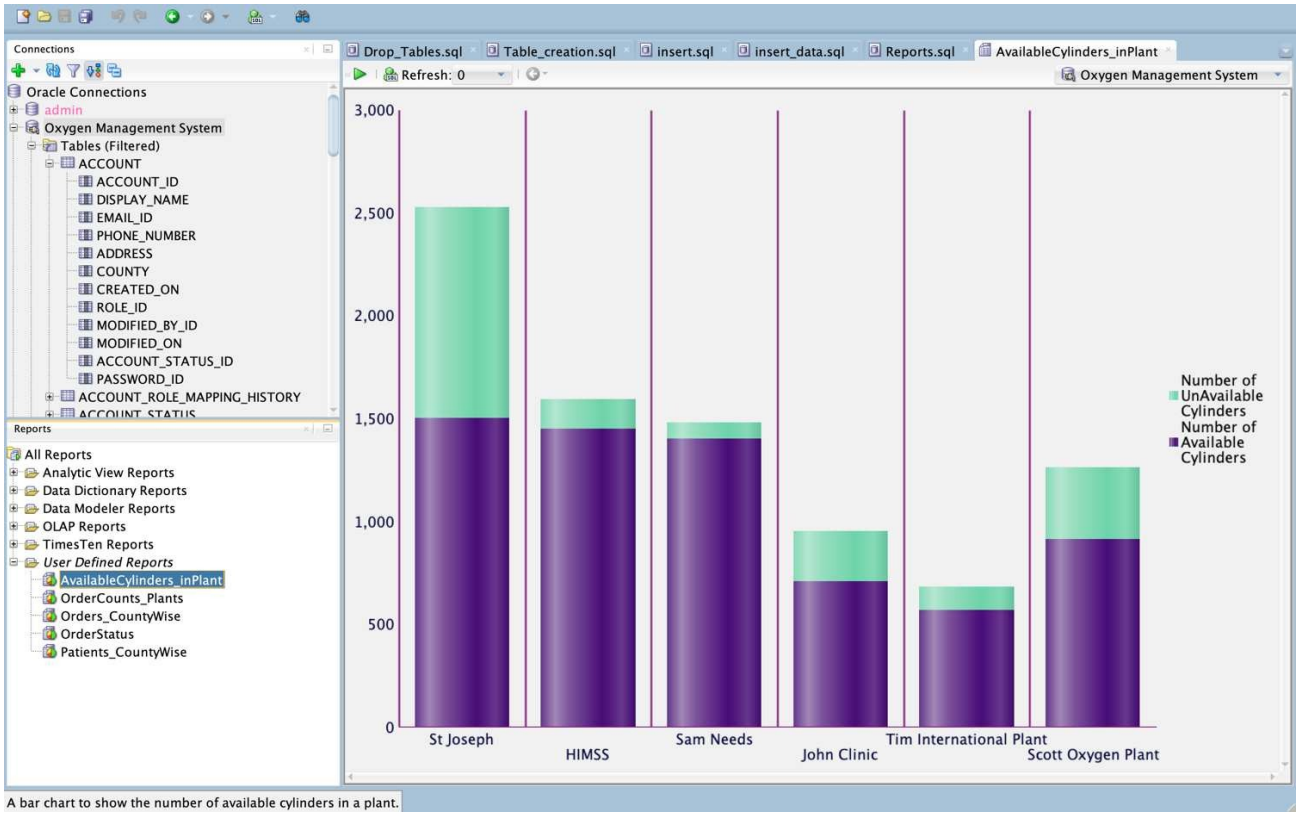
\

VISUALIZATION REPORTS USING QUERIES:

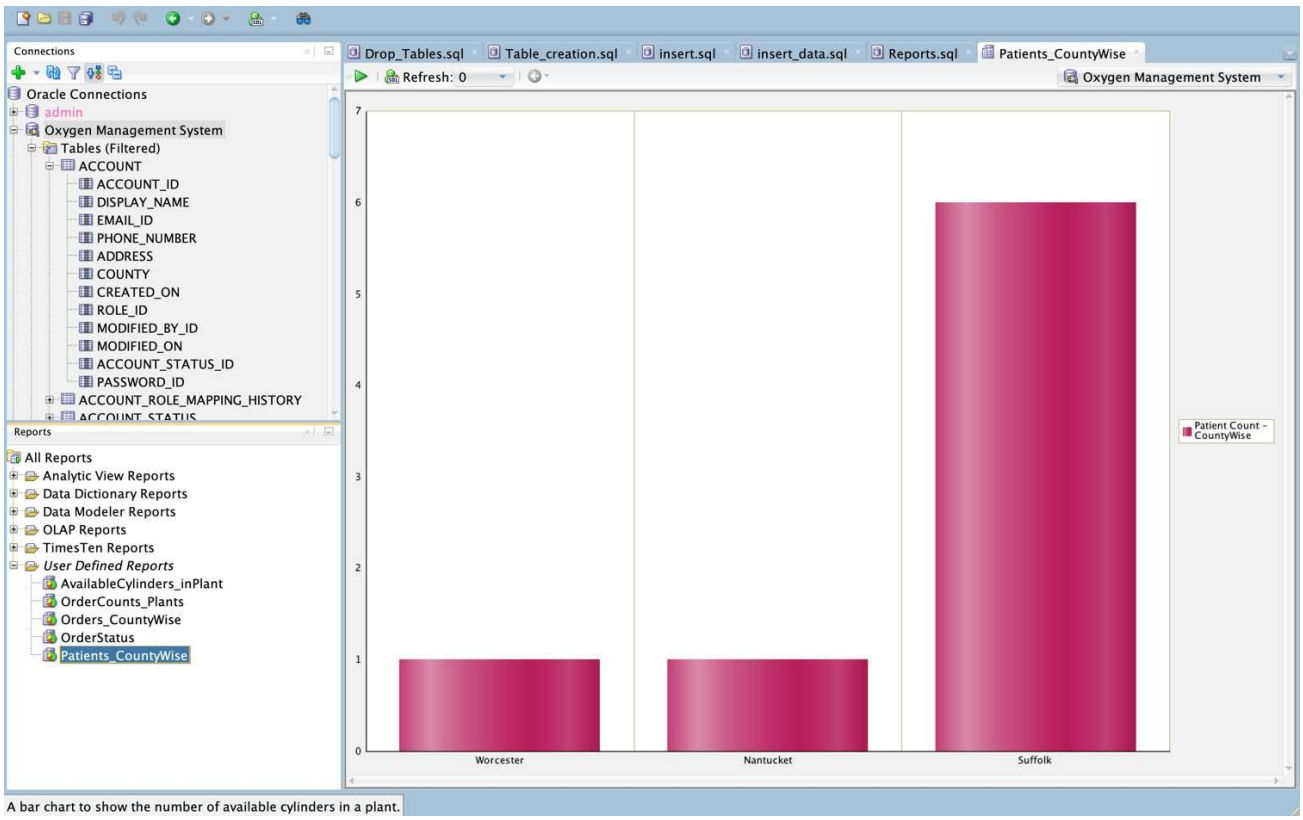
1. Count of orders raised at a Plant.



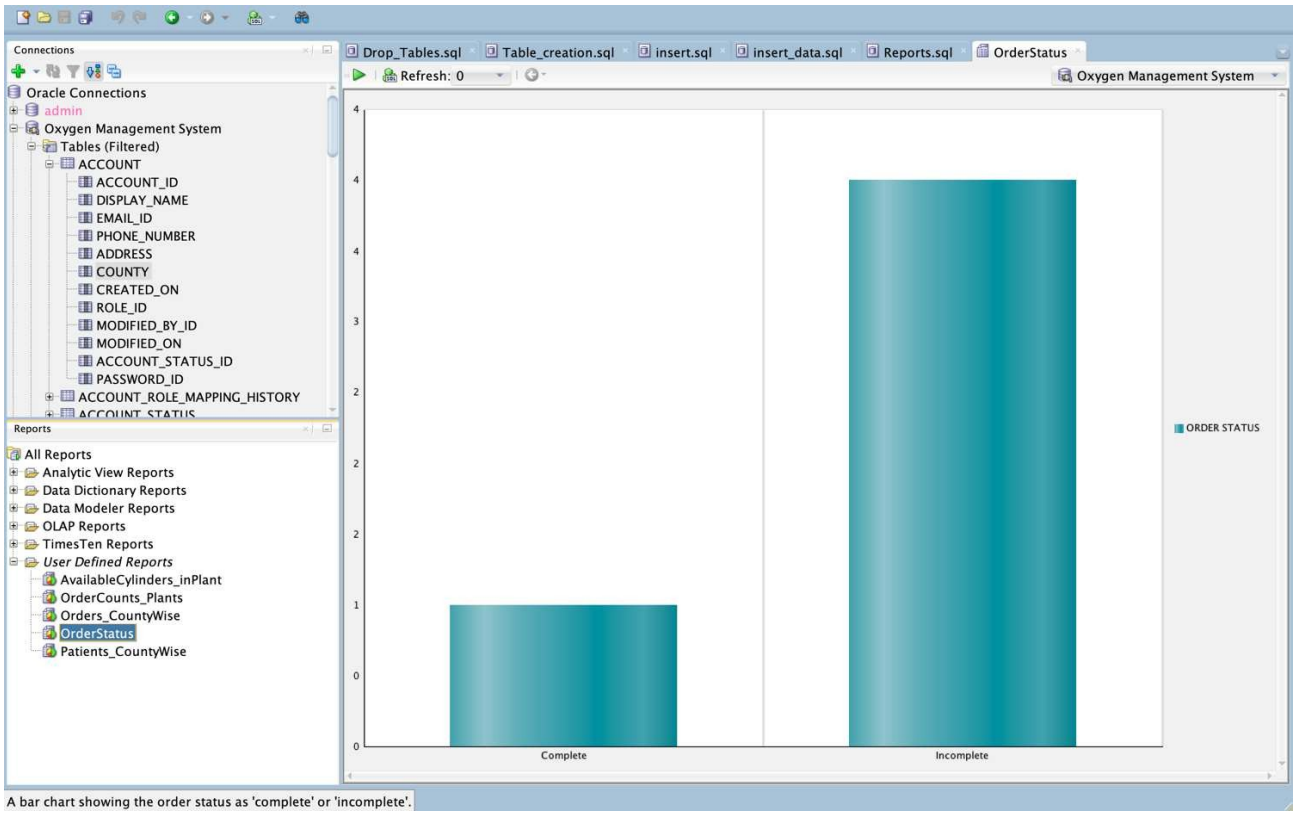
2. Number of Available and Unavailable Cylinders at a Plant.



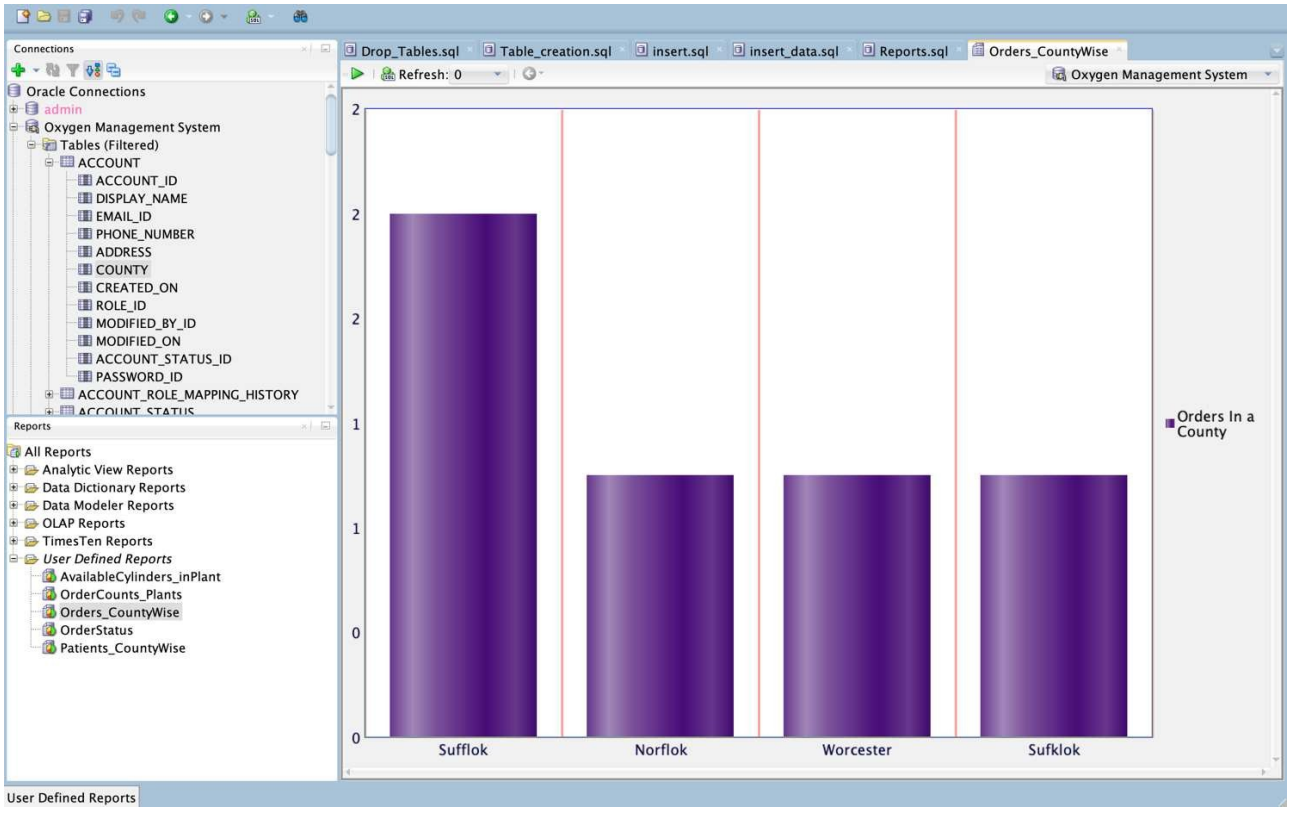
3. Number of patients in a county



4. Order Status : Complete (or) Incomplete

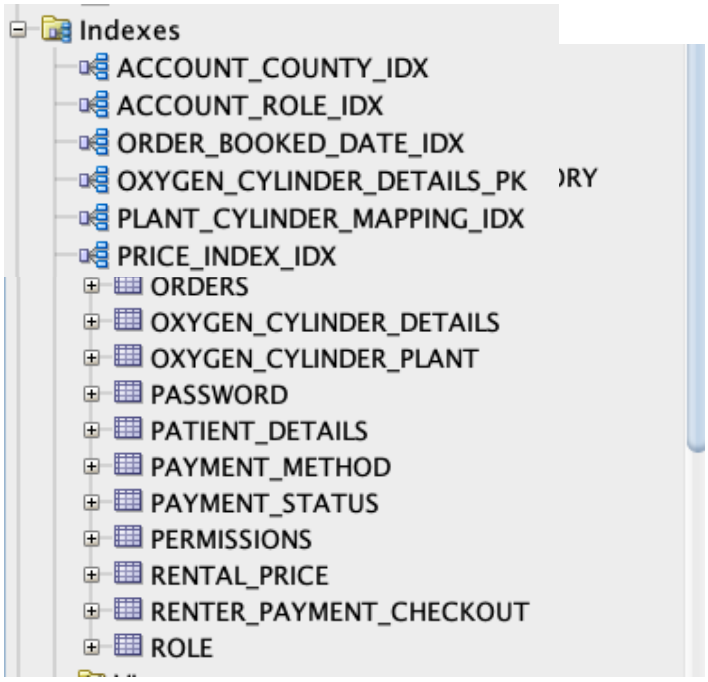


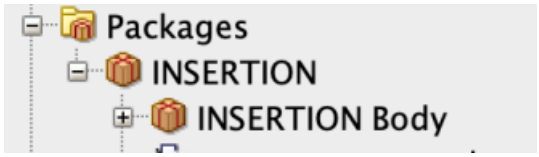
5. Order according to the county.

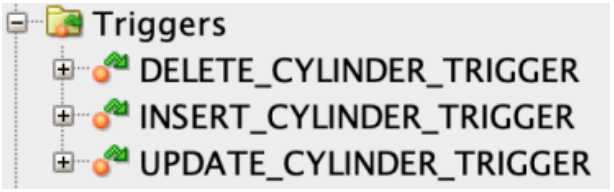


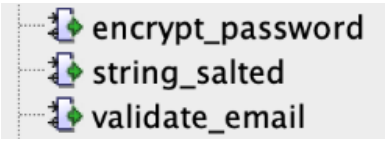
S.NO.	TOPIC	DESCRIPTION
1.	DDL (Create/ Drop)	Created tables using Stored Procedures.
2.	DML (Insert)	Stored Procedures - Package Insert - Functions (email, pwd encrypt, pwd salted) - Procedures (Insert with validations) - Exception Handling
3.	Indexes	To fetch values quicker.
4.	Triggers	Implementing manipulations of rows on Cylinder details table.
5.	Views & Reports	Created Views and respective user-defined reports in the form of charts.
6.	Test Cases	To justify the created validations of tables.

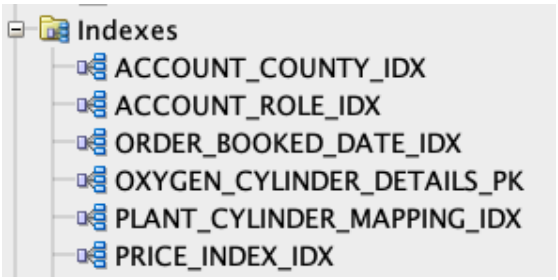
DATA OBJECTS:

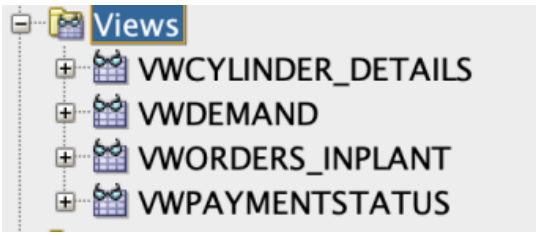












Grants:

```
--
DROP USER TEST_USER_2 CASCADE;
--
DECLARE
    user_name_to_create VARCHAR2(100);
BEGIN
        dbms_output.put_line('>>> Creating user using Procedure');
         user_name_to_create:='TEST_USER_2';
        EXECUTE IMMEDIATE 'create user ' || user_name_to_create || ' identified by
        dbms_output.put_line('>>> Allowing user to login');
        EXECUTE IMMEDIATE 'grant connect to ' || user_name_to_create ;
        dbms_output.put_line('>>> Allowing user to create a session and do ad-hoc')
        EXECUTE IMMEDIATE 'grant create session to ' || user_name_to_create;
        dbms_output.put_line('>>> Allowing user to create procedures');
        EXECUTE IMMEDIATE 'grant create procedure to ' || user_name_to_create;
        dbms_output.put_line('>>> Allowing user to create views');
        EXECUTE IMMEDIATE 'grant create view to ' || user_name_to_create ;
```

Script Output

Task completed in 7.184 seconds

```
Error starting at line : 3 in command -
DROP USER TEST_USER_1 CASCADE
Error report -
ORA-01940: cannot drop a user that is currently connected
01940. 00000 -  "cannot drop a user that is currently connected"
*Cause:    Attempt was made to drop a user that is currently logged in.
*Action:   Make sure user is logged off, then repeat command.

User TEST_USER_2 dropped.


PL/SQL procedure successfully completed.
```

Drop Table:

Worksheet Query Builder

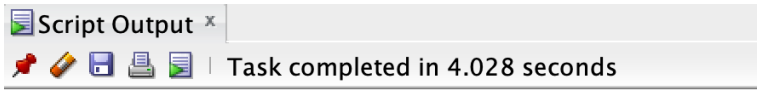
```
END;
/



BEGIN
        dbms_output.put_line('>>> Firing RENTER_PAYMENT_CHECKOUT t
        drop_table('RENTER_PAYMENT_CHECKOUT');
EXCEPTION
        WHEN OTHERS THEN
```

Script Output

Task completed in 1.352 seconds

```
Procedure DROP_TABLE compiled

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.
```

Insert Table:

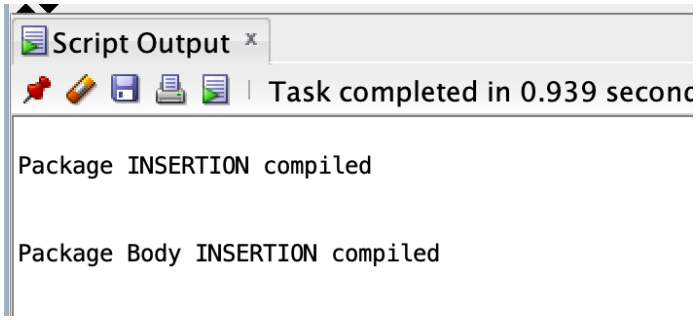



```
PL/SQL procedure successfully completed.

>>> Firing ACCOUNT Procedure
>>> Table ACCOUNT is created


PL/SQL procedure successfully completed.

Firing ACCOUNT ROLE MAPPING HISTORY Procedure
>>> Table ACCOUNT ROLE MAPPING HISTORY is created


PL/SQL procedure successfully completed.

>>> Firing AUTHENTICATION CONFIG Procedure
>>> Table AUTHENTICATION CONFIG is created


PL/SQL procedure successfully completed.

>>> Firing ORDERS Procedure
>>> Table ORDERS is created
```

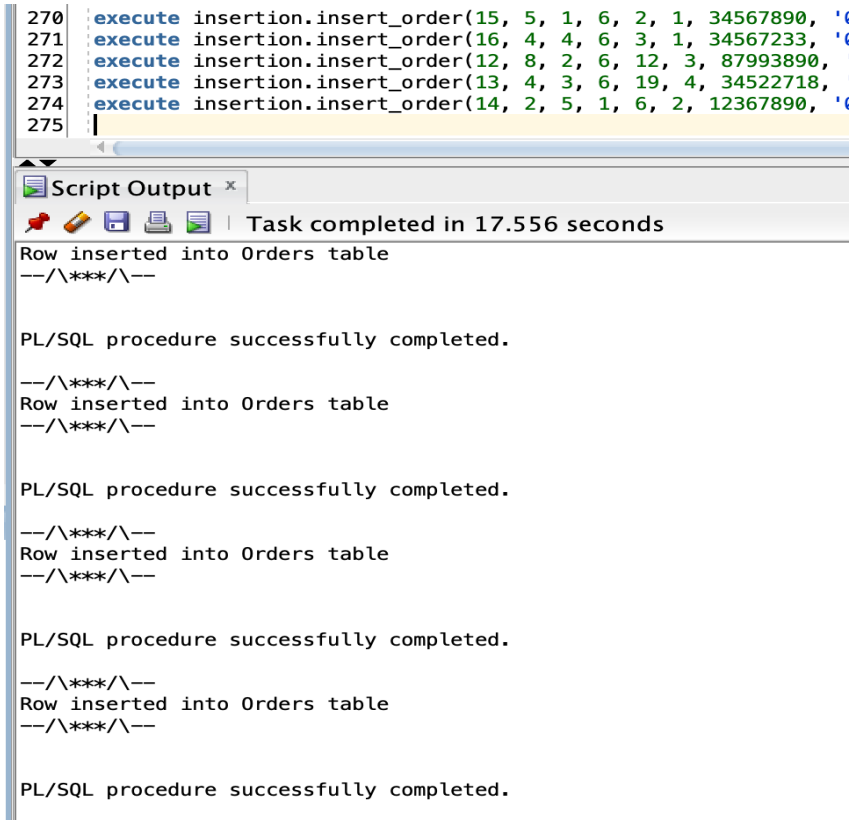
Index Creation:

```
PL/SQL procedure successfully completed.

Index created on price table successfully


PL/SQL procedure successfully completed.

Index created on Oxygen cylinder details table successfully


PL/SQL procedure successfully completed.

Index created on Account-id and County columns of Account table successfully


PL/SQL procedure successfully completed.

Index created on Account-id and Role-id columns of Account table successfully


PL/SQL procedure successfully completed.

Index created on Order-id and Booked Date columns of Orders table successfully


PL/SQL procedure successfully completed.
```

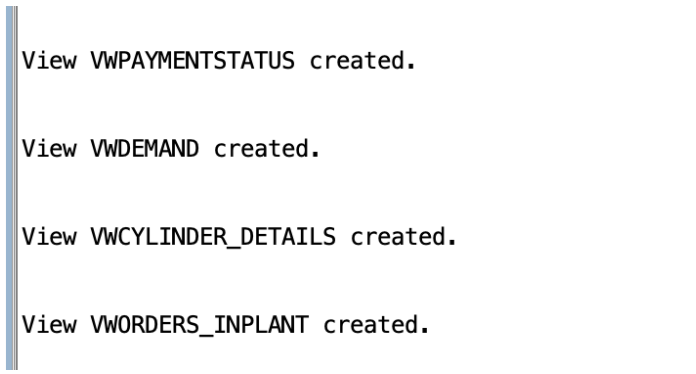
Insert Package and Procedures



Insert Data



Create Views



Create Triggers

```
70          raise_application_error('-20030', SQ
71  END;
72   /
73
```

Script Output x

Task completed in 0.544 second

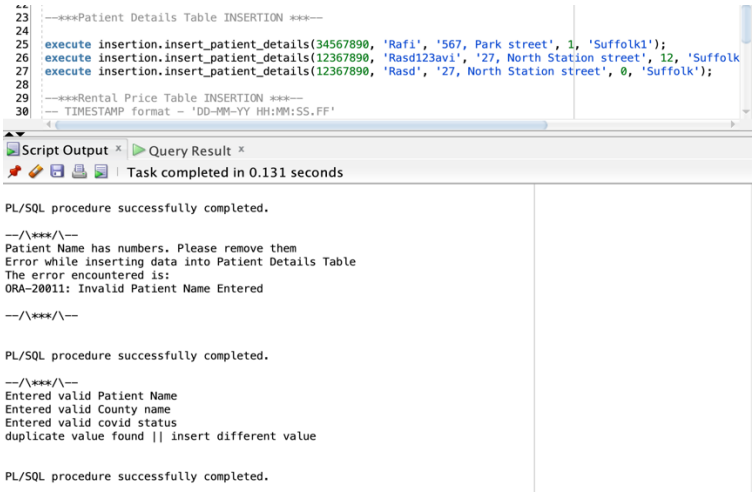
```
Trigger INSERT_CYLINDER_TRIGGER compiled

Trigger DELETE_CYLINDER_TRIGGER compiled

Trigger UPDATE_CYLINDER_TRIGGER compiled
```

Test-Cases

- execute insertion.insert_oxygen_plant_details('St Joseph', '134, Heath Street', 6176859693, 'Suffolk', 'gmailcom');

```
--/\***/\--
0
Entered valid Plant name
Entered valid County name
Error while inserting data into OXygen Cylidner Plant Table
The error encountered is:
ORA-20008: Invalid mail-id is entered

--/\***/\--


PL/SQL procedure successfully completed.
```

- execute insertion.insert_rental_price(TIMESTAMP '10-12-21 09:26:50.12', 100000);
- execute insertion.insert_rental_price(TIMESTAMP '9-12-21 09:26:50.12', 3);



Reports:

Worksheet Query Builder

```
1  -- REPORTS --
2
3  --- 1. View Payment Status ---
4
5  select paymentStatus, 'Orders_Count', count(numOfOrders)
6  from VwPaymentStatus group by paymentStatus;
7
8  --- 2. Demand in a county based on number of plants ----
9
10 select County_name, 'Number_of_Plants', count(plant_name) from VwDemand group by county_name;
11
12 ---- 3. Available cylinders in a plant ---
13
14 select Plant_Name,'Available_Cylinders', sum(numberOfUnits)
15 from VWcylinder_details
```

Script Output | Query Result

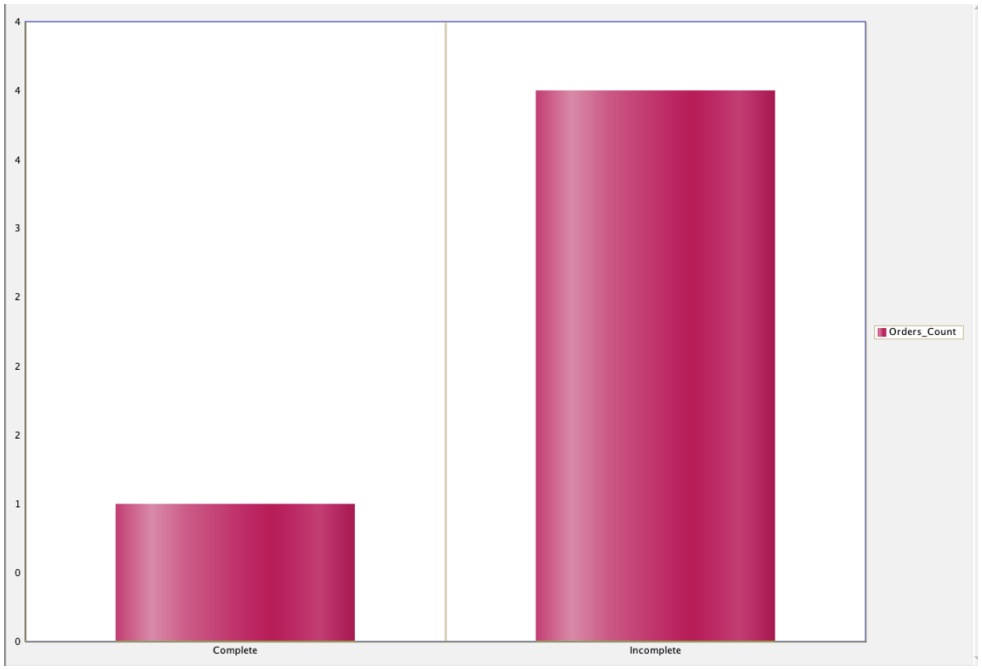
Task completed in 1.277 seconds

```
PAYMENTSTATUS
--------------------------------------------------------------------------------------------------------------------------------
'ORDERS_COUN COUNT(NUMOFORDERS)
------------ ------------------
Complete
Orders_Count                  1

Incomplete
Orders_Count                  4


COUNTY_NAME
--------------------------------------------------------------------------------------------------------------------------------
Worcester
Nantucket
Suffolk


PLANT_NAME
--------------------------------------------------------------------------------------------------------------------------------
John Clinic
HIMSS
Scott Oxygen Plant
Tim International Plant
St Joseph
Sam Needs
```

VIEW – USER DEFINED REPORTS:

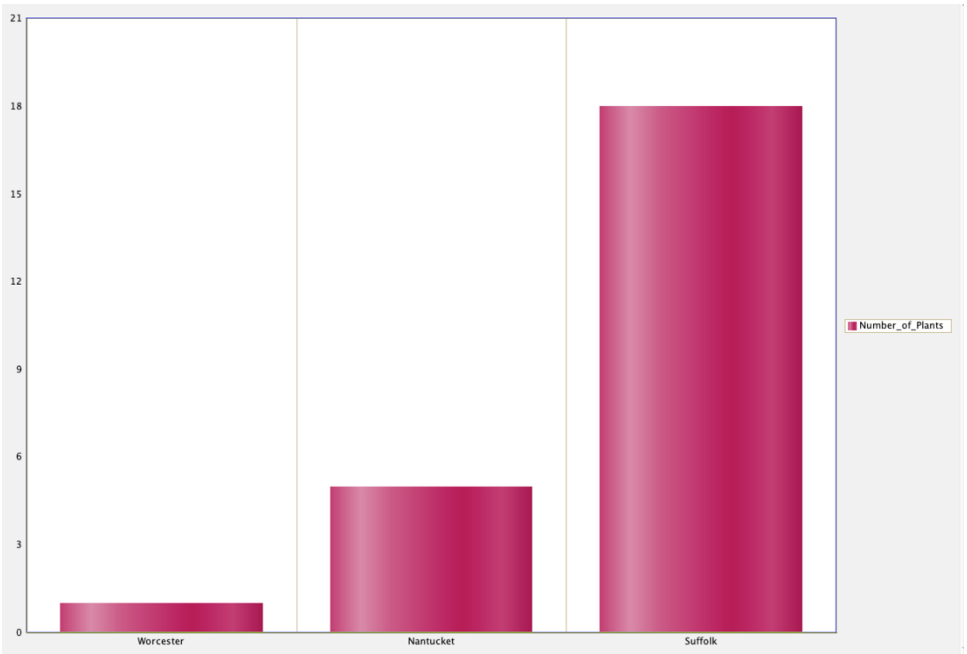
1. View Payment Status

```
select County_name, 'Number_of_Plants', count(plant_name)
from VwDemand group
```



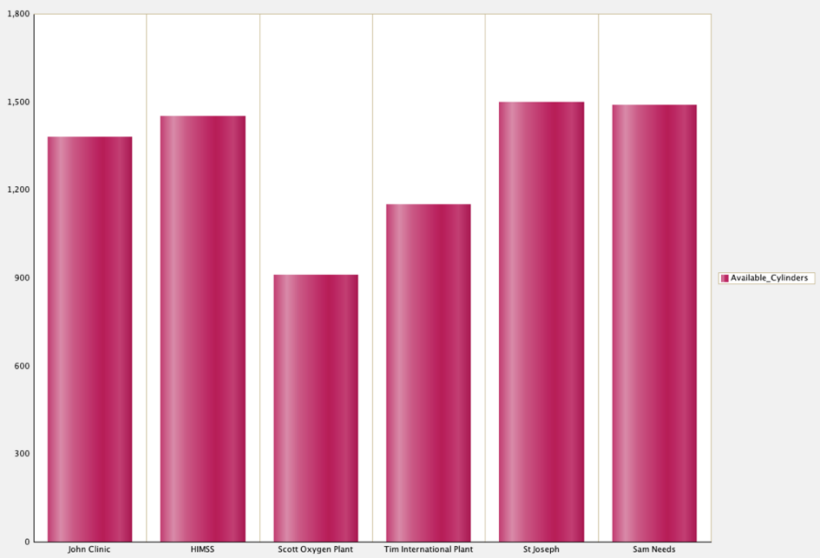
2. View Demand in a County

```
select County_name, 'Number_of_Plants', count(plant_name)
from VwDemand
group by county_name
```



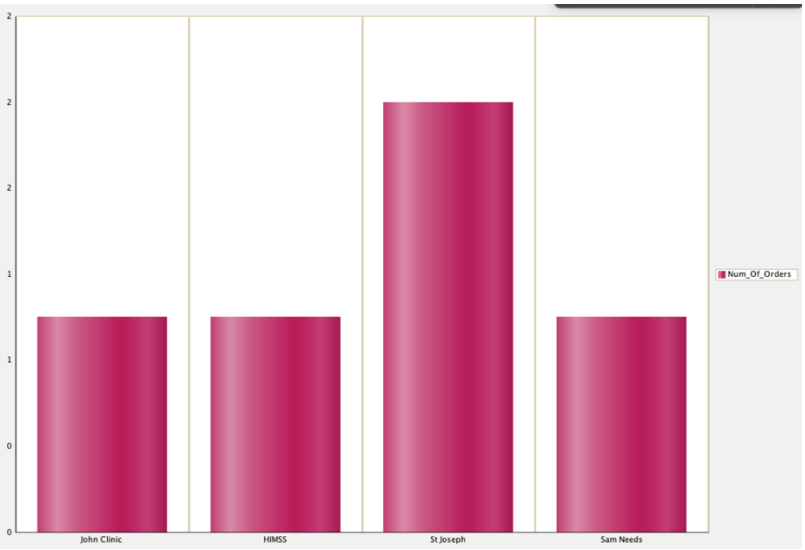
3. View Available Cylinders count in a plant.

```
select Plant_Name,'Available_Cylinders', sum(numberOfUnits)
from VWcylinder_details
WHERE availability = 1
Group by plant_name
```



4. View Orders placed to a plant.

```
select plant_name, 'Num_Of_Orders', count(OrderCount)
from VwOrders_InPlant
group by plant_name
```



Thank you!