

**INVENTORY STRUCTURE**

**CAPSTONE PROJECT-1**

**Panjal Chandrakant Yakswami**

**Vertocity | Data Science**

# **SQL:**

* Structured Query Language
* It is a standard language for Relational Database Management System.
* Database Management System is a platform where the user perform various operations
* It is used to store, manipulate and retrieve data.

# **TASK: 1(A)**

|  |
| --- |
| * **Create a database with name ‘INVENTORY’:** |

* **What is Database?**
  + Database is a place where the data is collected and organized properly.

**Syntax:**

|  |
| --- |
| CREATE DATABASE INVENTORY;  USE INVENTORY; |

# **TASK: 1(B)**

|  |
| --- |
| * **Create “PRODUCT”, “SUPPLIER”, “CUSTOMER”, “ORDERS” and “STOCK” table with all the specified constraints.** |

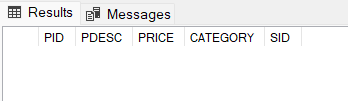
* **CREATE:**
  + It is a Data Definition Language (DDL).
  + Using DDL commands, it changes the structure of the table.
  + Create is used to make a table in the database.

1. **PRODUCT Table:**

* **QUERY:**

|  |
| --- |
| CREATE TABLE PRODUCT (  PID CHAR(5) PRIMARY KEY,  PDESC nVARCHAR(30) NOT NULL,  PRICE INT,  CATEGORY nVARCHAR(30),  SID CHAR(5)); |

* **RESULT:**



**NOTE:**

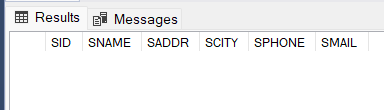
In order to provide Foreign Key in SID column of Product Table, first we need to create the Supplier Table and give SID column of Supplier Table as Primary Key.

1. **SUPPLIER Table:**

* **QUERY:**

|  |
| --- |
| CREATE TABLE SUPPLIER (  SID CHAR(5) PRIMARY KEY,  SNAME nVARCHAR(50) NOT NULL,  SADDR nVARCHAR(50) NOT NULL,  SCITY nVARCHAR(30) DEFAULT 'DELHI',  SPHONE BIGINT UNIQUE,  SMAIL nVARCHAR(50) UNIQUE); |

* **RESULT:**



* **Foreign Key for SID in PRODUCT Table:**

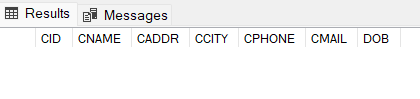
|  |
| --- |
| ALTER TABLE PRODUCT  ADD CONSTRAINT FK\_ID FOREIGN KEY (SID)  REFERENCES SUPPLIER(SID); |

1. **CUSTOMER Table:**

* **QUERY:**

|  |
| --- |
| CREATE TABLE CUSTOMER (  CID CHAR(5) PRIMARY KEY,  CNAME nVARCHAR(50) NOT NULL,  CADDR nVARCHAR(50) NOT NULL,  CCITY nVARCHAR(30) NOT NULL,  CPHONE BIGINT NOT NULL,  CMAIL nVARCHAR(50) NOT NULL,  DOB DATE CHECK(DOB < '01-01-2020')); |

* **RESULT:**

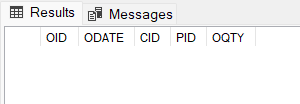


1. **ORDERS Table:**

* **QUERY:**

|  |
| --- |
| CREATE TABLE ORDERS (  OID CHAR(5) PRIMARY KEY,  ODATE DATE,  CID CHAR(5) REFERENCES CUSTOMER(CID),  PID CHAR(5) REFERENCES PRODUCT(PID),  OQTY INT CHECK(OQTY >=1)); |

* **RESULT:**

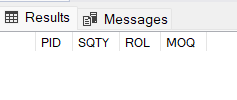


1. **STOCK Table:**

* **QUERY:**

|  |
| --- |
| CREATE TABLE STOCK (  PID CHAR(5) REFERENCES PRODUCT(PID),  SQTY INT CHECK (SQTY >=0),  ROL INT CHECK (ROL >0),  MOQ INT CHECK (MOQ >=5)); |

* **RESULT:**



* **Terms used while creating the above tables:**
  1. **Primary Key:**
     + This constraint is used to identify each row uniquely.
     + It is a combination of NUT NULL and UNIQUE constraints.
     + Each table can have only 1 Primary Key.
  2. **Foreign Key:**
     + Using this constraint, we can link two or more tables with each other.
     + It acts like a child table to other parent table.
     + It refers to primary key for other table.
  3. **Not Null:**
     + This constraint is used to make sure that the field value is not kept empty.
  4. **Unique:**
     + With the help of this constraint, we can avoid repetitive data and allows all rows to have unique data.
  5. **Check:**
     + It is used to check the condition applied and allows us to follow the condition being applied.
  6. **Default:**
     + Whenever a value is not given then by using this constraint we can automatically add the data by default.

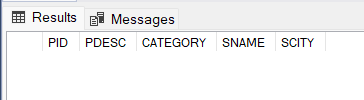
# **TASK: 2(A)**

|  |
| --- |
| * **Extract PID, PDESC, CATEGORY, SNAME and SCITY from the respective tables.** |

* **METHOD-1: Connecting Primary Key and Foreign Key**
* **QUERY:**

|  |
| --- |
| SELECT P.PID, P.PDESC,P.CATEGORY,S.SNAME,S.SCITY  FROM PRODUCT P, SUPPLIER S  WHERE P.SID = S.SID; |

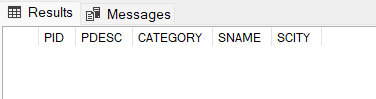
* **RESULT:**



* **METHOD-2: Using Join Method**
* **QUERY (INNER JOIN) :**

|  |
| --- |
| SELECT P.PID, P.PDESC,P.CATEGORY,S.SNAME,S.SCITY  FROM PRODUCT P  INNER JOIN SUPPLIER S  ON P.SID = S.SID; |

* **RESULT:**



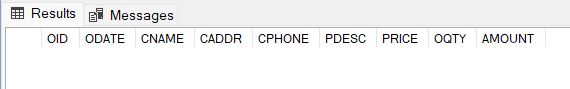
# **TASK: 2(B)**

|  |
| --- |
| * **Extract OID, ODATE, CNAME, CADDR, CPHONE, PDESC, PRICE, OQTY, AMOUNT.** |

* **METHOD-1: Connecting Primary Key and Foreign Key**
* **QUERY:**

|  |
| --- |
| SELECT O.OID, O.ODATE, C.CNAME,C.CADDR,C.CPHONE,P.PDESC,P.PRICE,  O.OQTY,AMOUNT=P.PRICE\*O.OQTY  FROM ORDERS O,CUSTOMER C,PRODUCT P  WHERE O.PID=P.PID AND O.CID=C.CID; |

* **RESULT:**



**NOTE:**

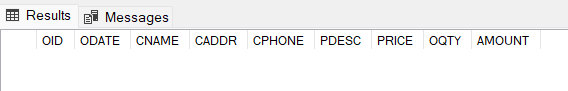
Here we need to add “AMOUNT” column using the relation between PRICE and OQTY.

AMOUNT= PRICE \* OQTY

* **METHOD-2: Using Join Method**
* **QUERY (INNER JOIN) :**

|  |
| --- |
| SELECT O.OID, O.ODATE,  C.CNAME,C.CADDR,C.CPHONE,P.PDESC,P.PRICE,  O.OQTY,AMOUNT=P.PRICE\*O.OQTY  FROM ORDERS O  INNER JOIN CUSTOMER C  ON O.CID=C.CID  INNER JOIN PRODUCT P  ON O.PID=P.PID; |

* **RESULT:**



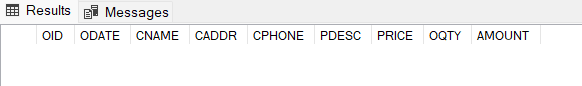
# **TASK: 2(C)**

|  |
| --- |
| * **Generate a view “BILL” that displays OID, ODATE, CNAME, CADDR, PHONE, PDESC, PRICE, OQTY and AMOUNT.** |

* **What is view?**
  + It is used to view any rows or columns depending on the requirement from the user.
  + It only displays the selected data from the table.
  + It can also add SQL statements and functions to view and present the data.
  + It is created with CREATE VIEW statement.
  + It is used for security purpose since they provide encapsulation of the name of table.
  + Data is not stored permanently.
* **QUERY:**

|  |
| --- |
| CREATE VIEW BILL  AS  SELECT O.OID, O.ODATE, C.CNAME, C.CADDR,  C.CPHONE,P.PDESC,P.PRICE,O.OQTY,AMOUNT=P.PRICE\*O.OQTY  FROM ORDERS O,CUSTOMER C,PRODUCT P  WHERE O.PID=P.PID AND O.CID=C.CID;  SELECT \* FROM BILL; |

* **RESULT:**



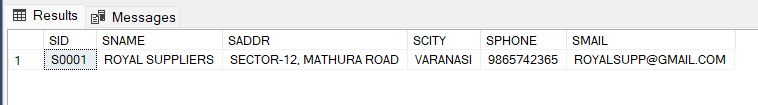
# **TASK: 3(A)**

|  |
| --- |
| * **Create simple procedure to ADDSUPPLIER to add details into “SUPPLIER” table and display the details of the newly added supplier.** |

* **What is Procedure?**
  + Procedure is a step-by-step process used to perform DML operations using the parameters passed in the procedure.
  + It enables reusability by passing same statements multiple times.
  + It can be easily modified, reusable and increases the performance.
* **QUERY:**

|  |
| --- |
| CREATE PROCEDURE ADDSUPPLIER (@I CHAR(5), @N nVARCHAR(50),  @A nVARCHAR(50), @C nVARCHAR(30), @P BIGINT, @M nVARCHAR(50))  AS  BEGIN  INSERT INTO SUPPLIER  VALUES (@I,@N,@A,@C,@P,@M);  SELECT \* FROM SUPPLIER;  END;  EXEC ADDSUPPLIER 'S0001', 'ROYAL SUPPLIERS', 'SECTOR-12, MATHURA ROAD', 'VARANASI', 9865742365, 'ROYALSUPP@GMAIL.COM'; |

* **RESULT:**

****

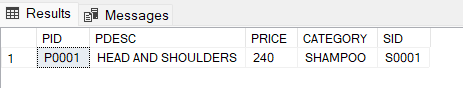
# **TASK: 3(B)**

|  |
| --- |
| * **Create simple procedure to ADDPRO to add details into “PRODUCT” table and display the details of the newly added product.** |

* **QUERY:**

|  |
| --- |
| CREATE PROCEDURE ADDPRO (@PI CHAR(5), @PD nVARCHAR(30),  @PP INT, @PC nVARCHAR(30), @PS CHAR(5))  AS  BEGIN  INSERT INTO PRODUCT  VALUES (@PI,@PD,@PP,@PC,@PS);  SELECT \* FROM PRODUCT;  END;  EXEC ADDPRO 'P0001', 'HEAD AND SHOULDERS', 240, 'SHAMPOO', 'S0001'; |

* **RESULT:**

****

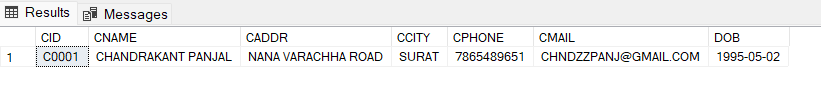
# **TASK: 3(C)**

|  |
| --- |
| * **Create simple procedure to ADDCUST to add details into “CUSTOMER” table and display the details of the newly added customer.** |

* **QUERY:**

|  |
| --- |
| CREATE PROCEDURE ADDCUST (@CI CHAR(5),  @CN nVARCHAR(50), @CA nVARCHAR(50), @CC nVARCHAR(30),  @CP BIGINT, @CM nVARCHAR(50), @CD DATE)  AS  BEGIN  INSERT INTO CUSTOMER  VALUES (@CI,@CN,@CA,@CC,@CP,@CM,@CD);  SELECT \* FROM CUSTOMER;  END;  EXEC ADDCUST 'C0001', 'CHANDRAKANT PANJAL',  'NANA VARACHHA ROAD', 'SURAT', 7865489651,  'CHNDZZPANJ@GMAIL.COM', '05-02-1995'; |

* **RESULT:**

****

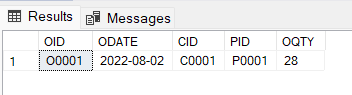
# **TASK: 3(D)**

|  |
| --- |
| * **Create simple procedure to ADDORDER to add details into “ORDERS” table and display the details of the newly added orders. ODATE should be generated automatically as the current date.** |

* **QUERY:**

|  |
| --- |
| CREATE PROCEDURE ADDORDER (@OI CHAR(5),  @OC CHAR(5), @OP CHAR(5), @OQ INT)  AS  BEGIN  INSERT INTO ORDERS  VALUES (@OI,GETDATE(),@OC,@OP,@OQ);  SELECT \* FROM ORDERS;  END;  EXEC ADDORDER 'O0015','C0001','P0001',28; |

* **RESULT:**

****

# **TASK: 4(A)**

|  |
| --- |
| * **Create function to auto generate 5-character alpha numeric ID that accepts 2 parameters which holds a character and the number. The function should return the ID by concatenating the character, required zeroes and the specific number.** |

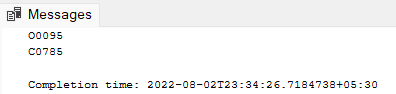
* **What is Function?**
  + Function is a process of storing any program in which user gives some input parameters to get a desired output.
  + Using functions, there can be different kind of results:
    - * 1. Scalar Function gives single result.
        2. Tabular Function gives result in table form.
* **QUERY:**

|  |
| --- |
| CREATE FUNCTION AUTOID (@C CHAR(1),@N INT)  RETURNS CHAR(5)  AS  BEGIN  DECLARE @ID AS CHAR(5)=CASE  WHEN @N<10 THEN CONCAT(@C,'000',@N)  WHEN @N<100 THEN CONCAT(@C,'00',@N)  WHEN @N<1000 THEN CONCAT(@C,'0',@N)  WHEN @N<10000 THEN CONCAT(@C,@N)  END;  RETURN @ID  END; |

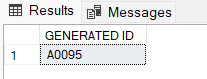
* **INPUT:**

|  |
| --- |
| PRINT DBO.AUTOID ('O',95);  PRINT DBO.AUTOID('C',785); |
| SELECT DBO.AUTOID('A',95) AS 'GENERATED ID'; |

* **RESULT-1:**

****

* **RESULT-2:**

****

# **TASK: 4(B)**

|  |
| --- |
| * **Drop and recreate the procedures in which the ID should be automatically created using the above function and new sequence.** |

* **QUERY TO DROP THE PROCEDURES:**

|  |
| --- |
| DROP PROCEDURE ADDSUPPLIER;  DROP PROCEDURE ADDPRO;  DROP PROCEDURE ADDCUST;  DROP PROCEDURE ADDORDER; |

* **What is Sequence?**
  + Sequenceis a process of generating numeric values automatically and to produce unique values.
  + It is used to avoid repeatability and maintain uniqueness in the data.

1. **SUPPLIER Table:**

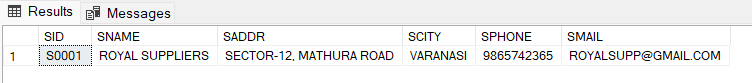
* **QUERY TO CREATE SEQUENCE FOR SUPPLIER TABLE:**

|  |
| --- |
| CREATE SEQUENCE SNUM  AS INT  START WITH 1  INCREMENT BY 1; |

* **QUERY TO CREATE PROCEDURE FOR SUPPLIER TABLE BY GENERATING SID AUTOMATICALLY:**

|  |
| --- |
| CREATE PROCEDURE ADDSUPPLIER (@N nVARCHAR(50),  @A nVARCHAR(50), @C nVARCHAR(30), @P BIGINT,  @M nVARCHAR(50))  AS  BEGIN  DECLARE @I INT  DECLARE @ID CHAR(5)  SET @I = (NEXT VALUE FOR SNUM)  SET @ID = DBO.AUTOID ('S',@I)  INSERT INTO SUPPLIER  VALUES (@ID,@N,@A,@C,@P,@M);  SELECT \* FROM SUPPLIER;  END;  EXEC ADDSUPPLIER 'ROYAL SUPPLIERS', 'SECTOR-12, MATHURA ROAD',  'VARANASI', 9865742365, 'ROYALSUPP@GMAIL.COM'; |

* **RESULT:**

****

1. **PRODUCT Table:**

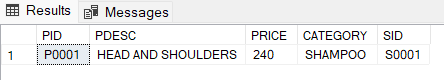
* **QUERY TO CREATE SEQUENCE FOR PRODUCT TABLE:**

|  |
| --- |
| CREATE SEQUENCE PNUM  AS INT  START WITH 1  INCREMENT BY 1; |

* **QUERY TO CREATE PROCEDURE FOR PRODUCT TABLE BY GENERATING PID AUTOMATICALLY:**

|  |
| --- |
| CREATE PROCEDURE ADDPRO (@PD nVARCHAR(30),  @PP INT, @PC nVARCHAR(30), @PS CHAR(5))  AS  BEGIN  DECLARE @I INT  DECLARE @ID CHAR(5)  SET @I = (NEXT VALUE FOR PNUM)  SET @ID = DBO.AUTOID ('P',@I)  INSERT INTO PRODUCT  VALUES (@ID,@PD,@PP,@PC,@PS);  SELECT \* FROM PRODUCT;  END;  EXEC ADDPRO 'HEAD AND SHOULDERS', 240, 'SHAMPOO', 'S0001'; |

* **RESULT:**

****

1. **CUSTOMER Table:**

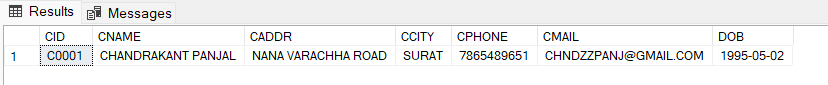
* **QUERY TO CREATE SEQUENCE FOR CUSTOMER TABLE:**

|  |
| --- |
| CREATE SEQUENCE CNUM  AS INT  START WITH 1  INCREMENT BY 1; |

* **QUERY TO CREATE PROCEDURE FOR CUSTOMER TABLE BY GENERATING CID AUTOMATICALLY:**

|  |
| --- |
| CREATE PROCEDURE ADDCUST (@CN nVARCHAR(50),  @CA nVARCHAR(50) ,@CC nVARCHAR(30),  @CP BIGINT, @CM nVARCHAR(50), @CD DATE)  AS  BEGIN  DECLARE @I INT  DECLARE @ID CHAR(5)  SET @I = (NEXT VALUE FOR CNUM)  SET @ID = DBO.AUTOID ('C',@I)  INSERT INTO CUSTOMER  VALUES (@ID,@CN,@CA,@CC,@CP,@CM,@CD);  SELECT \* FROM CUSTOMER;  END;  EXEC ADDCUST 'CHANDRAKANT PANJAL',  'NANA VARACHHA ROAD', 'SURAT', 7865489651,  'CHNDZZPANJ@GMAIL.COM','05-02-1995'; |

* **RESULT:**

****

1. **ORDERS Table:**

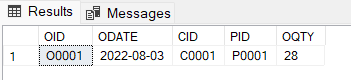
* **QUERY TO CREATE SEQUENCE FOR ORDERS TABLE:**

|  |
| --- |
| CREATE SEQUENCE ONUM  AS INT  START WITH 1  INCREMENT BY 1; |

* **QUERY TO CREATE PROCEDURE FOR ORDERS TABLE BY GENERATING OID AUTOMATICALLY:**

|  |
| --- |
| CREATE PROCEDURE ADDORDER (@OC CHAR(5),  @OP CHAR(5), @OQ INT)  AS  BEGIN  DECLARE @I INT  DECLARE @ID CHAR(5  SET @I = (NEXT VALUE FOR ONUM)  SET @ID = DBO.AUTOID ('O',@I)  INSERT INTO ORDERS  VALUES (@ID,GETDATE(),@OC,@OP,@OQ);  SELECT \* FROM ORDERS;  END;  EXEC ADDORDER 'C0001','P0001',28; |

* **RESULT:**

****

**THANK YOU…**