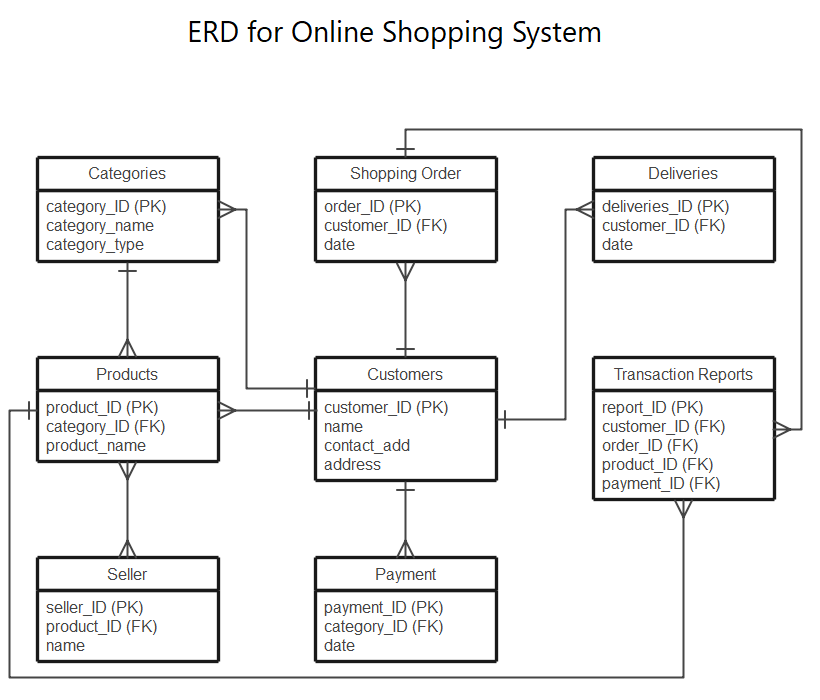
**SQL Assignment 2**

**Q1) For an online purchasing database, create entity relationship diagrams. Create a database object from your entity diagram.**

****

**Q2) Create a SQL store process to register the use of the database, complete it with proper validation and transaction rollback and commit.**

|  |
| --- |
| \*/ |
|  | create procedure spupdatesalary |
|  | as |
|  | begin |
|  | begin try |
|  | begin tran |
|  | update [dbo].[tblemployee] set salary = 9000 where id = 1 |
|  | commit tran |
|  | end try |
|  | begin catch |
|  | rollback tran |
|  | end catch |
|  | end |
|  |  |
|  | exec spupdatesalary |

**Q3) List the SQL aggregate function and demonstrate how to utilize it.**

* COUNT FUNCTION

COUNT function is used to Count the number of rows in a database table. It can work on both numeric and non-numeric data types. COUNT function uses the COUNT (\*) that returns the count of all the rows in a specified table. COUNT (\*) considers duplicate and Null.

* SUM Function

Sum function is used to calculate the sum of all selected columns. It works on numeric fields only.

* AVG function

The AVG function is used to calculate the average value of the numeric type. AVG function returns the average of all non-Null values.

* MAX Function

MAX function is used to find the maximum value of a certain column. This function determines the largest value of all selected values of a column.

* MIN Function

MIN function is used to find the minimum value of a certain column. This function determines the smallest value of all selected values of a column.

**Q4) In SQL, create a pivot query.**

* PIVOT in SQL Server  
  PIVOT relational operator converts data from row level to column level. PIVOT rotates a table-valued expression by turning the unique values from one column in the expression into multiple columns in the output. Using PIVOT operator, we can perform aggregate operation where we need them.

Syntax:

1. Pivot:

SELECT (ColumnNames)

FROM (TableName)

PIVOT

(

AggregateFunction (ColumnToBeAggregated)

FOR PivotColumn IN (PivotColumnValues)

) AS

**Q**5) **With an example, describe how to join in SQL.**

* SQL join statements allow us to access information from two or more tables at once. They also keep our database normalized. Normalization allows us to keep data redundancy low so that we can decrease the amount of data anomalies in our application when we delete or update a record. Let’s use the example above with our customer and the customer’s order to illustrate. If we had a customer’s table that had information about our customer and a separate orders table:Table

  Description automatically generated
* In these tables, take notice that there is a lot of the same information in both tables. A join statement greatly reduces the need for these duplicate values. Our new tables could look like this:

Table, treemap chart

Description automatically generated

* Types of Join statements

The type of join statement you use depends on your use case. There are four different types of join operations:

* **(INNER) JOIN:** Returns dataset that have matching values in both tables
* **LEFT (OUTER) JOIN:** Returns all records from the left table and matched records from the rights
* **RIGHT (OUTER) JOIN:** Returns all records from the right table and the matched records from the left
* **FULL (OUTER) JOIN:** Returns all records when there is a match in either the left table or right table

**Q6) How to locate the 4th highest value in a column in a row. Create your table.**

* select SAL from EMPLOYEE E1 where (4 - 1) = (select count (distinct (SAL)) from EMPLOYEEE2 where E2.SAL > E1.SAL)

use [H+Active]

select \* from [dbo].[ProductListing]

where [Price] =

(

select MIN([Price]) from [dbo].[ProductListing]

where [Price] in

(select distinct top 4 [Price] from [dbo].[ProductListing]

order by [Price] desc)

)