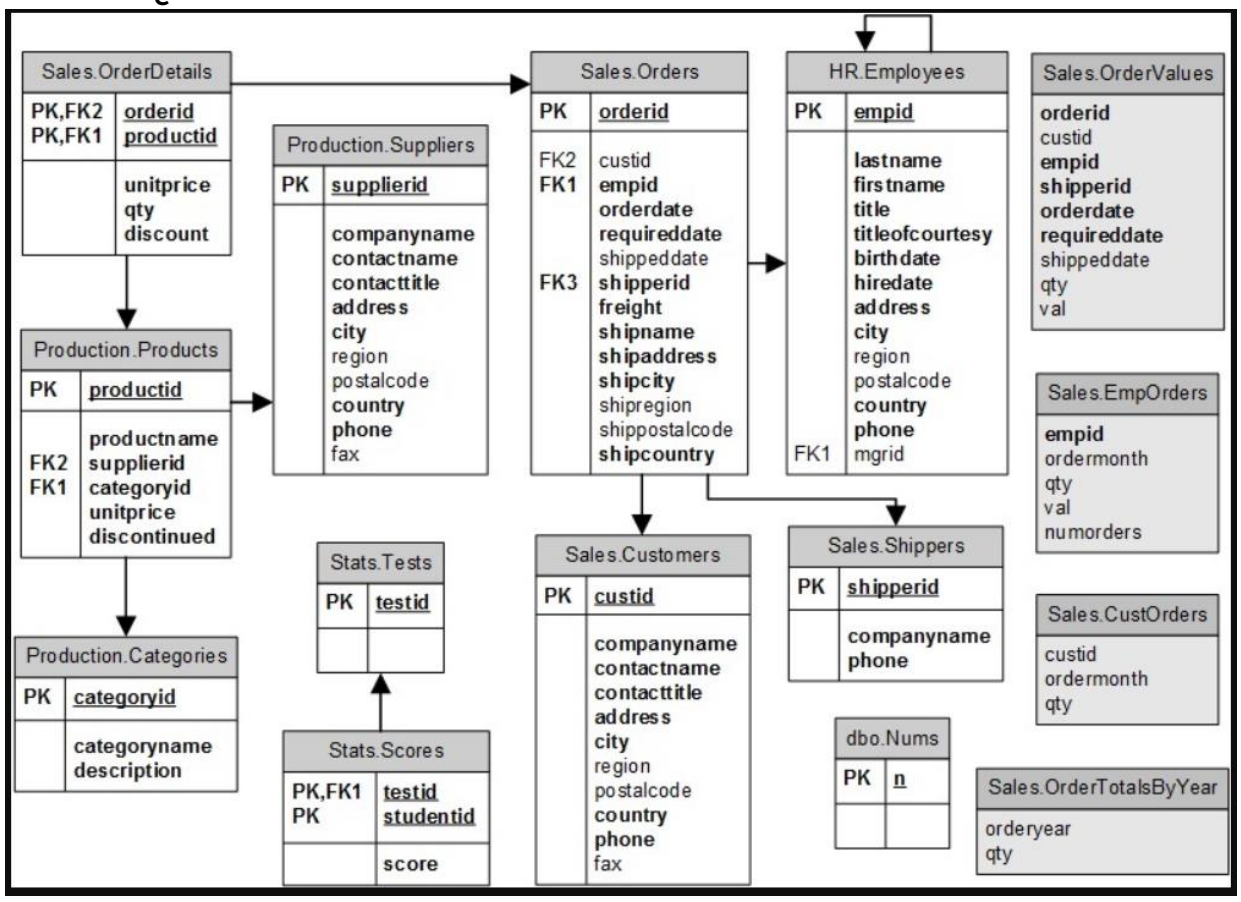
```
/*
Neba Nfonsang
Project 4: Queries that use set operations
such as UNION, INTERSECT or EXCEPT.
*/

USE TSQLV4
```




```
*/
```

Query # 1: Use UNION ALL to unify two input query results without removing duplicates
***/**

```
SELECT country, region, city
FROM HR.Employees

UNION ALL

SELECT country, region, city
FROM Sales.Customers;
```

/*
Query # 2: Use UNION to unify the results of two queries and eliminate duplicates
***/**

```
SELECT country, region, city
FROM HR.Employees

UNION

SELECT country, region, city
FROM Sales.Customers;
```

/*

```
Query # 3: Use INTERSECT operator to return distinct
rows that appear in both input query results
*/

SELECT country, region, city
FROM HR.Employees

INTERSECT

SELECT country, region, city
FROM Sales.Customers;

/*
Query # 4: Use the ROW_NUM function to get
intersection of duplicates as well since INTERSECT
ALL is not implemented directly in T-SQL
*/

SELECT
    ROW_NUMBER()
        OVER(PARTITION BY country, region, city
             ORDER BY (SELECT 0)) AS rownum,
    country, region, city
FROM HR.Employees

INTERSECT

SELECT
    ROW_NUMBER()
        OVER(PARTITION BY country, region, city
             ORDER BY (SELECT 0)),
    country, region, city
FROM Sales.Customers;
```

```
/*
Query # 5: Exclude the row number attribute from the
results
*/
WITH INTERSECT_ALL
AS
 (
    SELECT
        ROW_NUMBER()
        OVER(PARTITION BY country, region, city
                  ORDER BY (SELECT 0)) AS rownum,
        country, region, city
    FROM HR.Employees

    INTERSECT

    SELECT
        ROW_NUMBER()
        OVER(PARTITION BY country, region, city
                  ORDER BY (SELECT 0)),
        country, region, city
    FROM Sales.Customers
 )
SELECT country, region, city
FROM INTERSECT_ALL;
```

```
/*
```

```
Query # 6: Use EXCEPT to return the difference in
query inputs
- Return rows in first query that are not in the
second query
- Return records on Employee table that are not on
Customer table
*/

SELECT country, region, city
FROM HR.Employees

EXCEPT

SELECT country, region, city
FROM Sales.Customers;

/*
Query # 7: Return records on Customer table that are
not in Employee table
*/

SELECT country, region, city
FROM Sales.Customers

EXCEPT

SELECT country, region, city
FROM HR.Employees;

/*
Query # 8: To consider duplicates differences, use
the ROW_NUMBER function
*/
WITH INTERSECT_ALL
```

```
AS
 (
    SELECT
        ROW_NUMBER()
        OVER(PARTITION BY country, region, city
                  ORDER BY (SELECT 0)) AS rownum,
        country, region, city
    FROM HR.Employees

    EXCEPT

    SELECT
        ROW_NUMBER()
        OVER(PARTITION BY country, region, city
                  ORDER BY (SELECT 0)),
        country, region, city
    FROM Sales.Customers
 )
SELECT country, region, city
FROM INTERSECT_ALL;

/*
Query # 9: Precedence: INTERSECT preceeds UNION and
EXCEPT UNION and EXCEPT are evaluated according to
the order in which they appear
*/

SELECT country, region, city
FROM Production.Suppliers

EXCEPT

SELECT country, region, city
FROM HR.Employees
```

```
INTERSECT

SELECT country, region, city
FROM Sales.Customers;

/*
Query # 10: Use parenthesis to control the order of
precedence
*/

(SELECT country, region, city
FROM Production.Suppliers

EXCEPT

SELECT country, region, city
FROM HR.Employees)

INTERSECT

SELECT country, region, city
FROM Sales.Customers;

/*
Extra Query # 11:
*/

SELECT country, COUNT(*) AS numlocations
FROM (SELECT country, region, city FROM HR.Employees
      UNION
      SELECT country, region, city FROM
Sales.Customers) AS U
GROUP BY country;
```