Alejandro Monterrubio Navarro

Héctor Martín Cuadra

# Memoria Práctica 1 EDAT

payments (**checknumber**, **customernumber->customers.customernumber**, payment, amount)

customers (**customernumber**, customername, contactlastname, contactfirstname, phone, addressline1, addressline2, city, state, postalcode, country, creditlimit, salesremployeenumber->employees.employeenumber)

employees (**employeenumber**, lastname, firstname, extension, email, jobtitle, reportsto->employees.employeenumber, officecode->offices.officecode)

offices (**officecode**, city, phone, addressline1, addressline2, state, country, postalcode, territory)

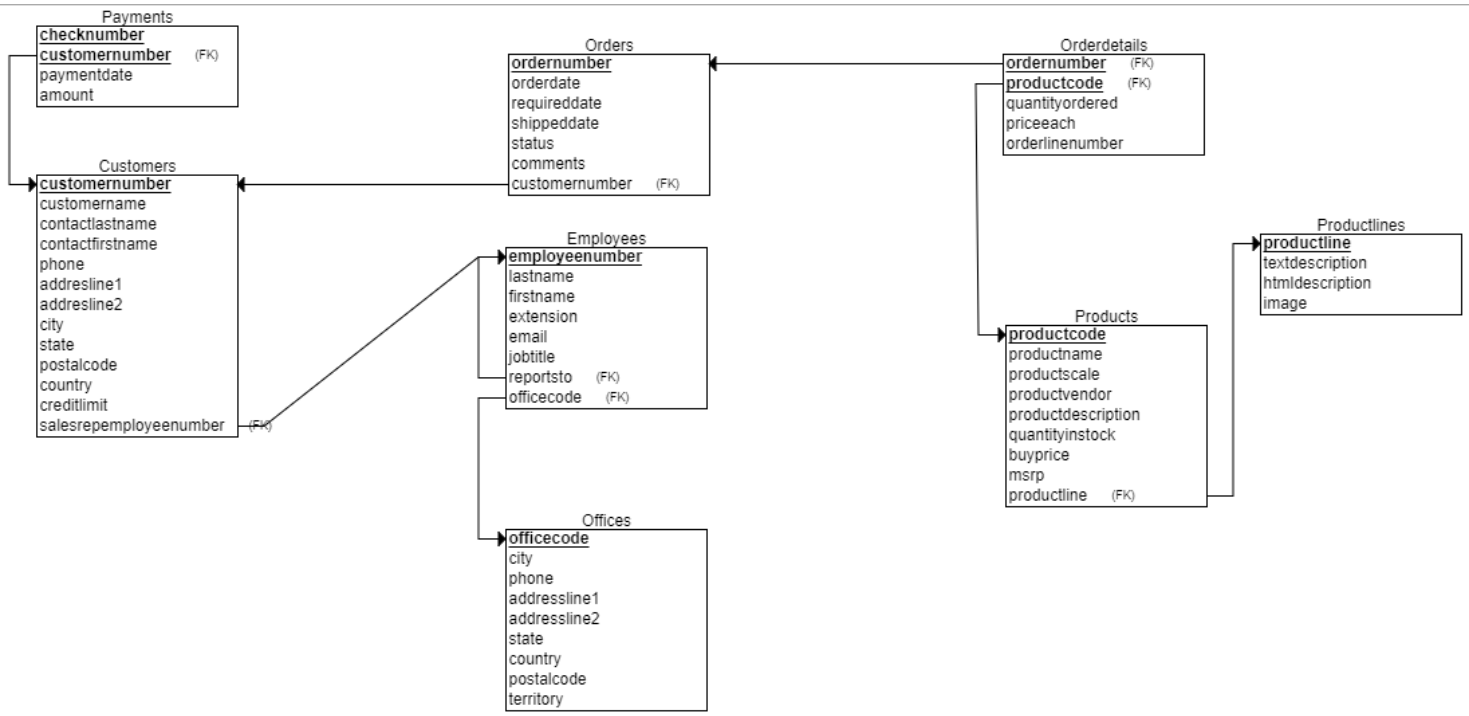
orders (**ordernumber**, orderdate, requireddate, shippeddate, status, comments, customernumber->customers.customernumber)

orderdetails (**ordernumber->orders.ordernumber**, **productcode->products.productcode**, quantityordered, priceeach, orderlinenumber)

products (**productcode**, productname, productscale, productvendor, productdescription, quantityinstock, buyprice, msrp, productline->poroductsline.productline)

productslines (**productline**, textdescription, htmldescription, image)

## Diagrama:



## Query 1:

Se utiliza select distinct para que devuelva los diferentes valores que involucren la compra del 1940 Ford Pickup Truck. Se agrupa con "group by" por customernumber y customername y se ordena en descendiente con "order by". El resultado al que se llama "total" se obtiene haciendo el "sum" de p2.buyprice para que muestre el valor total y no cada uno por separado.

```
SELECT c.customernumber,
       c.customername,
       Sum (p2.buyprice) AS total
FROM   customers AS c,
       orders o,
       orderdetails o2,
       products p2
WHERE  o.customernumber = c.customernumber
       AND o2.ordernumber = o.ordernumber
       AND o2.productcode = p2.productcode
       AND c.customernumber IN (SELECT DISTINCT c.customernumber
                                FROM   customers AS c,
                                       orders o,
                                       orderdetails o2,
                                       products p2
                                WHERE  o.customernumber = c.customernumber
                                       AND o2.ordernumber = o.ordernumber
                                       AND o2.productcode = p2.productcode
                                       AND p2.productname =
                                           '1940 Ford Pickup Truck')
GROUP  BY c.customernumber,
          c.customername
ORDER  BY total DESC
```

## Query 2:

Se hace select de productiline y la media (avg -> average) de shippedate - orderdate, a la que se llama "DiasDiferencia", la cual se mostrará en el resultado junto a productline. Tras establecer las relaciones entre tablas en "where" de orders, orderdetails, productlines y products, y finalmente se agrupa con "group by" por p2.productline.

```
SELECT p2.productline,
       Avg(o2.shippeddate - o2.orderdate) AS DiasDiferencia
FROM   productlines AS p2,
       orders AS o2,
       orderdetails o,
       products p
WHERE  o2.ordernumber = o.ordernumber
       AND o.productcode = p.productcode
       AND p.productline = p2.productline
GROUP  BY p2.productline
```

## Query 3:

Se hace uso de la tabla employee, hacemos select del número, apellido y reports del empleado y se hace uso de subconsultas que busquen aquellos empleados que reporten al director, y un nivel por encima, se busca a aquellos empleados que reporten a empleados con la restricción de que estos últimos sean los que se encuentren en la última subconsulta, es decir los que reporten al director.

```
SELECT e4.employeenumber,
       e4.lastname,
       e4.reportsto
FROM   employees e4
WHERE  e4.reportsto IN (SELECT e2.employeenumber
                        FROM   employees e2
                        WHERE  e2.reportsto IN (SELECT e3.employeenumber
                                                FROM   employees e3
                                                WHERE  e3.reportsto IS NULL))
```

## Query 4:

Hemos juntado el código de las oficinas con sus empleados y las ventas que han hecho estos luego hemos sumado la cantidad de productos que se han pedido y hemos creado una tabla con los códigos de las oficinas y el número de productos vendidos ordenados de manera decreciente.

```
SELECT o.officecode,
       Sum(o3.quantityordered) AS numberOfSoldItems
FROM   offices o
       NATURAL JOIN employees e
       JOIN customers c
         ON c.salesrepemployeenumber = e.employeenumber
       NATURAL JOIN orders o2
       NATURAL JOIN (SELECT o3.ordernumber,
                            Sum(o3.quantityordered) AS quantityordered
                     FROM   orderdetails o3
                     GROUP  BY o3.ordernumber) AS o3
GROUP  BY o.officecode
ORDER  BY numberofsolditems DESC
LIMIT  1;
```

## Query 5:

Para esta query hemos seleccionado el nombre de los países con el número de oficinas en estos y luego hemos cogido todas las oficinas (haciendo uniones entre código de oficina, empleado y las ventas a los clientes) que no estuvieran dentro de cualquier venta realizada durante el 2003.

```
SELECT o2.country,
       Count(*) AS numberOfOffice
FROM   offices o2
WHERE  o2.officecode NOT IN (SELECT e.officecode
                             FROM   employees e
                             WHERE  e.employeenumber IN
                                    (SELECT c.salesrepemployeenumber
                                     FROM   customers c
                                     WHERE  c.customernumber IN
                                            (SELECT o.customernumber
                                             FROM   orders o
                                             WHERE
                                            Extract(
                                            year FROM o.orderdate)
                                            = '2003' )))
GROUP  BY o2.country
ORDER  BY numberofoffice DESC;
```

## Query 6:

Seleccionamos los códigos de los productos, y para evitar que los productos no salgan repetidos con las id (ID1 y ID2 = ID2 y ID1) lo que hacemos es poner la condición `o1.productcode < o2.productcode` para que así el código que sería 1 y 2 = 2 y 1 solo pueda coger el casi 2 y 1 y elimine la otra posibilidad para no repetirla. Finalmente los emparejamos y contamos las parejas de productos, luego se muestran en orden decreciente.

```
SELECT o1.productcode,
       o2.productcode,
       Count(*) AS n
FROM   orderdetails o1,
       orderdetails o2
WHERE  o1.ordernumber = o2.ordernumber
       AND o1.productcode < o2.productcode
GROUP  BY o1.productcode,
          o2.productcode
HAVING Count(*) > 1
ORDER  BY n DESC;
```