# Data Analysis and Integration

Introduction to data warehousing

# A customer order
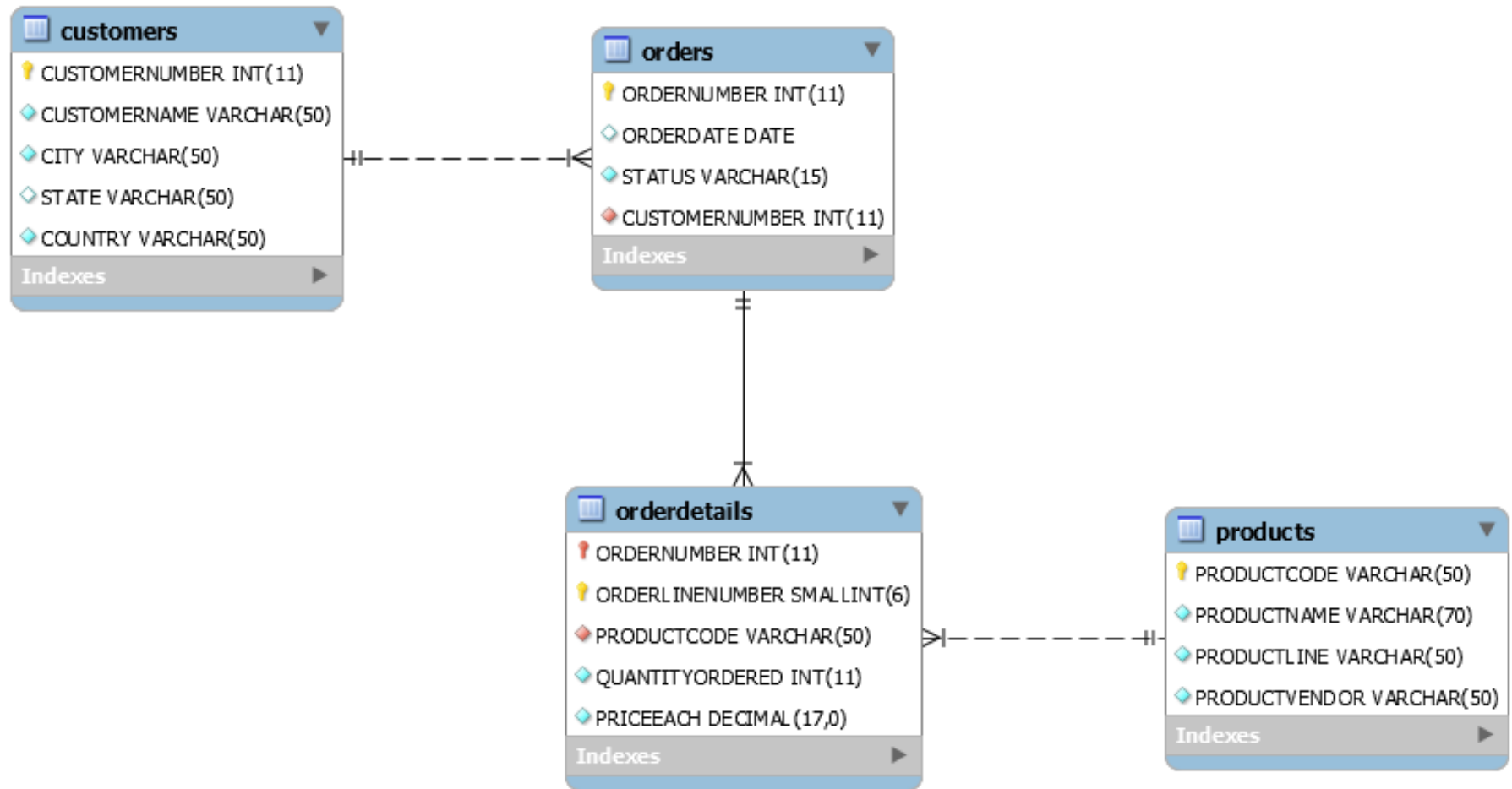
**Order number:** 10100
**Order date:** 2003-01-06

**Customer:**
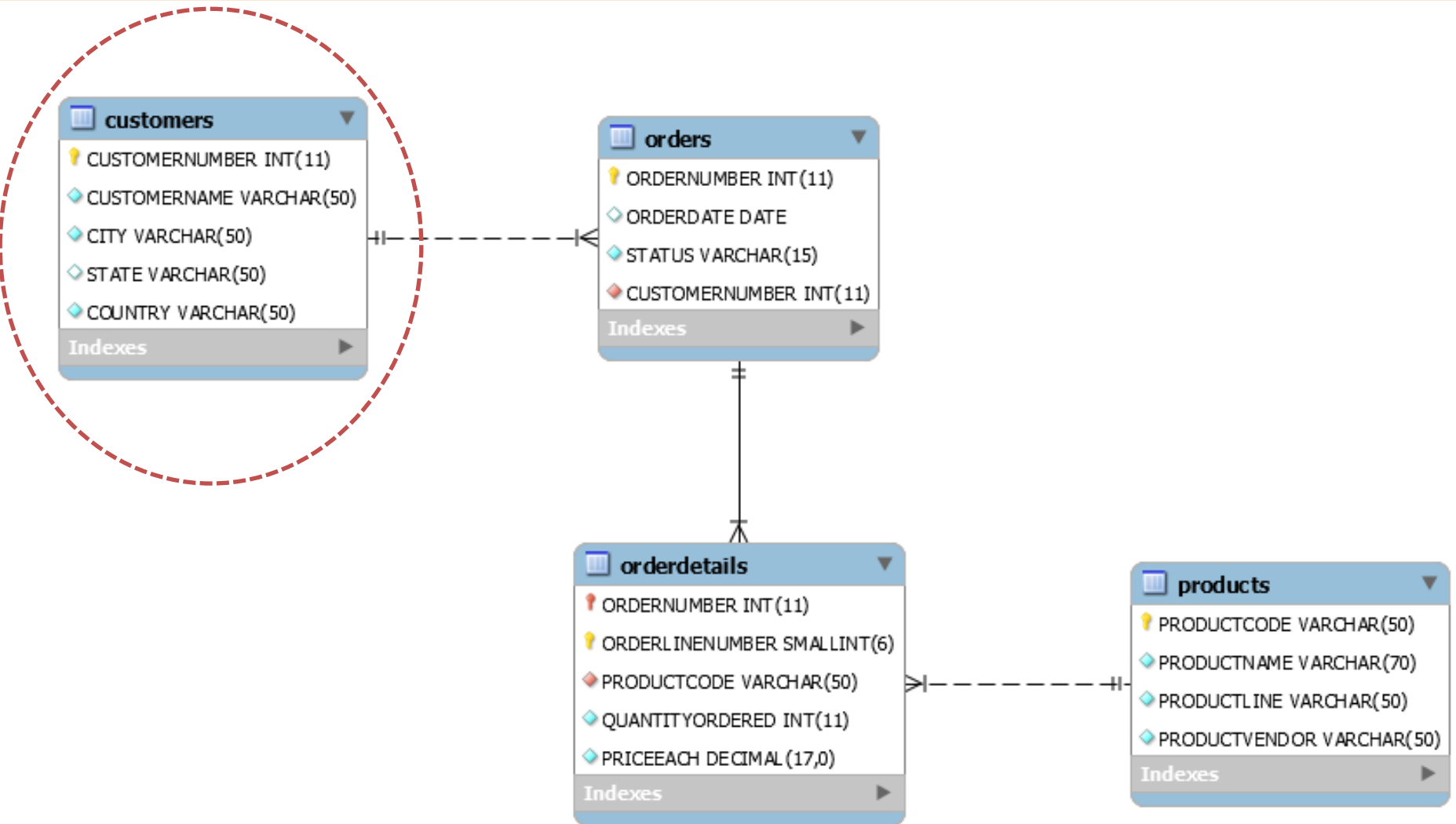Toys of Finland, Co.
Keskuskatu 45
Helsinki, Finland

| Line | Product | Quantity | Unit price | Total |
|------|---------|----------|------------|-------|
| 1 | 1936 Mercedes Benz 500k Roadster | 49 | 34 | 1666 |
| 2 | 1911 Ford Town Car | 50 | 68 | 3400 |
| 3 | 1917 Grand Touring Sedan | 30 | 172 | 5160 |
| 4 | 1932 Alfa Romeo 8C2300 Spider Sport | 22 | 87 | 1914 |

Grand total: 12140

# The steelwheels database

# The customers table

# The customers table

| CUSTOMERNUMBER | CUSTOMERNAME | CITY | STATE | COUNTRY |
|---|---|---|---|---|
| 97 | Madison Inc | ST  AUGUSTINE | FL | USA |
| 98 | Johnson Inc | ST Cloud | FL | USA |
| 99 | Tarallo Inc | Sanford | FL | USA |
| 100 | Audio Video 'R' Us | Orlando | FL | USA |
| 103 | Atelier graphique | Nantes | NULL | France |
| 112 | Signal Gift Stores | Las Vegas | NV | USA |
| 114 | Australian Collectors, Co. | Melbourne | Victoria | Australia |
| 119 | La Rochelle Gifts | Nantes | NULL | France |
| 121 | Baane Mini Imports | Stavern | NULL | Norway |
| 124 | Mini Gifts Distributors Ltd. | San Rafael | CA | USA |
| 125 | Havel & Zbyszek Co | Warszawa | NULL | Poland |
| 128 | Blauer See Auto, Co. | Frankfurt | NULL | Germany |
| 129 | Mini Wheels Co. | San Francisco | CA | USA |
| 131 | Land of Toys Inc. | NYC | NY | USA |
| 141 | Euro+ Shopping Channel | Madrid | NULL | Spain |
| 144 | Volvo Model Replicas, Co | Luleå | NULL | Sweden |
| 145 | Danish Wholesale Imports | Kobenhavn | NULL | Denmark |
| 146 | Saveley & Henriot, Co. | Lyon | NULL | France |
| 148 | Dragon Souveniers, Ltd. | Singapore | NULL | Singapore |
| 151 | Muscle Machine Inc | NYC | NY | USA |

# The orders table
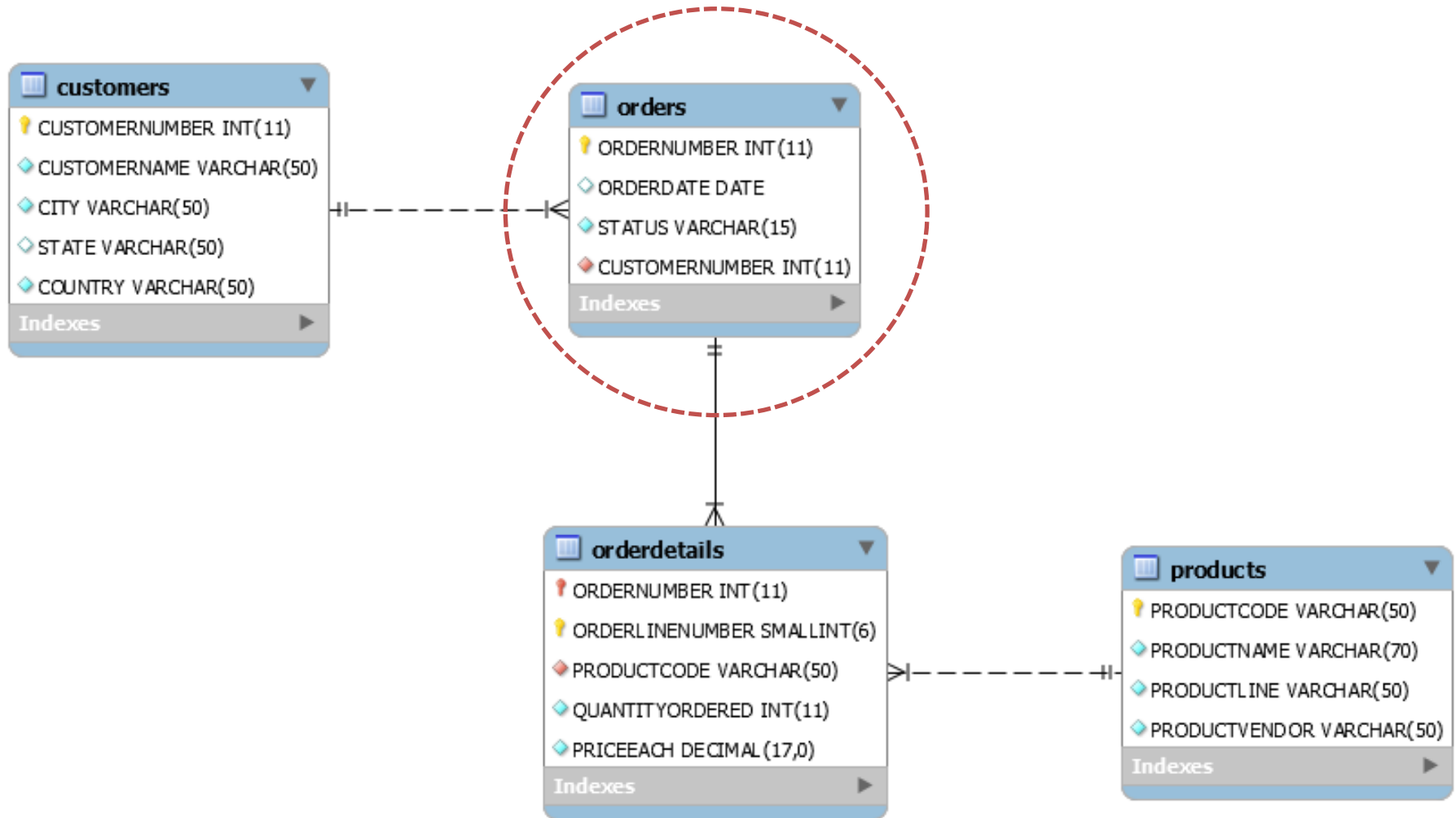
# The orders table

```
+-------------+------------+---------+----------------+
| ORDERNUMBER | ORDERDATE  | STATUS  | CUSTOMERNUMBER |
+-------------+------------+---------+----------------+
|       10100 | 2003-01-06 | Shipped |            363 |
|       10101 | 2003-01-09 | Shipped |            128 |
|       10102 | 2003-01-10 | Shipped |            181 |
|       10103 | 2003-01-29 | Shipped |            121 |
|       10104 | 2003-01-31 | Shipped |            141 |
|       10105 | 2003-02-11 | Shipped |            145 |
|       10106 | 2003-02-17 | Shipped |            278 |
|       10107 | 2003-02-24 | Shipped |            131 |
|       10108 | 2003-03-03 | Shipped |            385 |
|       10109 | 2003-03-10 | Shipped |            486 |
|       10110 | 2003-03-18 | Shipped |            187 |
|       10111 | 2003-03-25 | Shipped |            129 |
|       10112 | 2003-03-24 | Shipped |            144 |
|       10113 | 2003-03-26 | Shipped |            124 |
|       10114 | 2003-04-01 | Shipped |            172 |
|       10115 | 2003-04-04 | Shipped |            424 |
|       10116 | 2003-04-11 | Shipped |            381 |
|       10117 | 2003-04-16 | Shipped |            148 |
|       10118 | 2003-04-21 | Shipped |            216 |
|       10119 | 2003-04-28 | Shipped |            382 |
+-------------+------------+---------+----------------+
```

# The orderdetails table
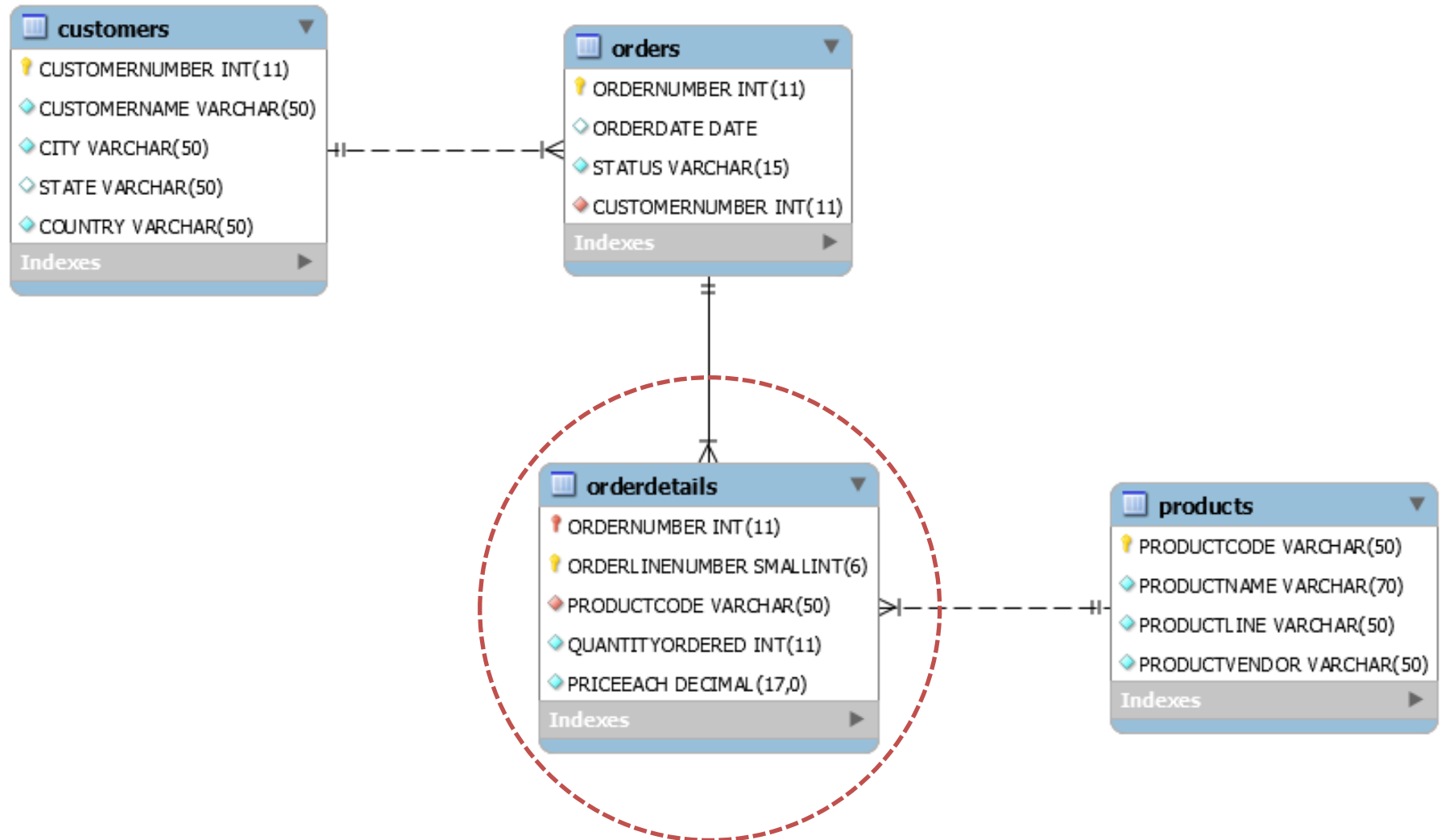
# The orderdetails table

```
+------------+-----------------+-------------+-----------------+-----------+
| ORDERNUMBER | ORDERLINENUMBER | PRODUCTCODE | QUANTITYORDERED | PRICEEACH |
+------------+-----------------+-------------+-----------------+-----------+
|      10100 |               1 | S24_3969    |              49 |        34 |
|      10100 |               2 | S18_2248    |              50 |        68 |
|      10100 |               3 | S18_1749    |              30 |       172 |
|      10100 |               4 | S18_4409    |              22 |        87 |
|      10101 |               1 | S18_2795    |              26 |       145 |
|      10101 |               2 | S24_2022    |              46 |        54 |
|      10101 |               3 | S24_1937    |              45 |        31 |
|      10101 |               4 | S18_2325    |              25 |       151 |
|      10102 |               1 | S18_1367    |              41 |        50 |
|      10102 |               2 | S18_1342    |              39 |       123 |
|      10103 |               1 | S24_2300    |              36 |       102 |
|      10103 |               2 | S18_2432    |              22 |        54 |
|      10103 |               3 | S32_1268    |              31 |       104 |
|      10103 |               4 | S10_4962    |              42 |       129 |
|      10103 |               5 | S18_4600    |              36 |       117 |
|      10103 |               6 | S700_2824   |              42 |       106 |
|      10103 |               7 | S32_3522    |              45 |        76 |
|      10103 |               8 | S12_1666    |              27 |       126 |
|      10103 |               9 | S18_4668    |              41 |        47 |
|      10103 |              10 | S18_1097    |              35 |       112 |
+------------+-----------------+-------------+-----------------+-----------+
```

# The products table
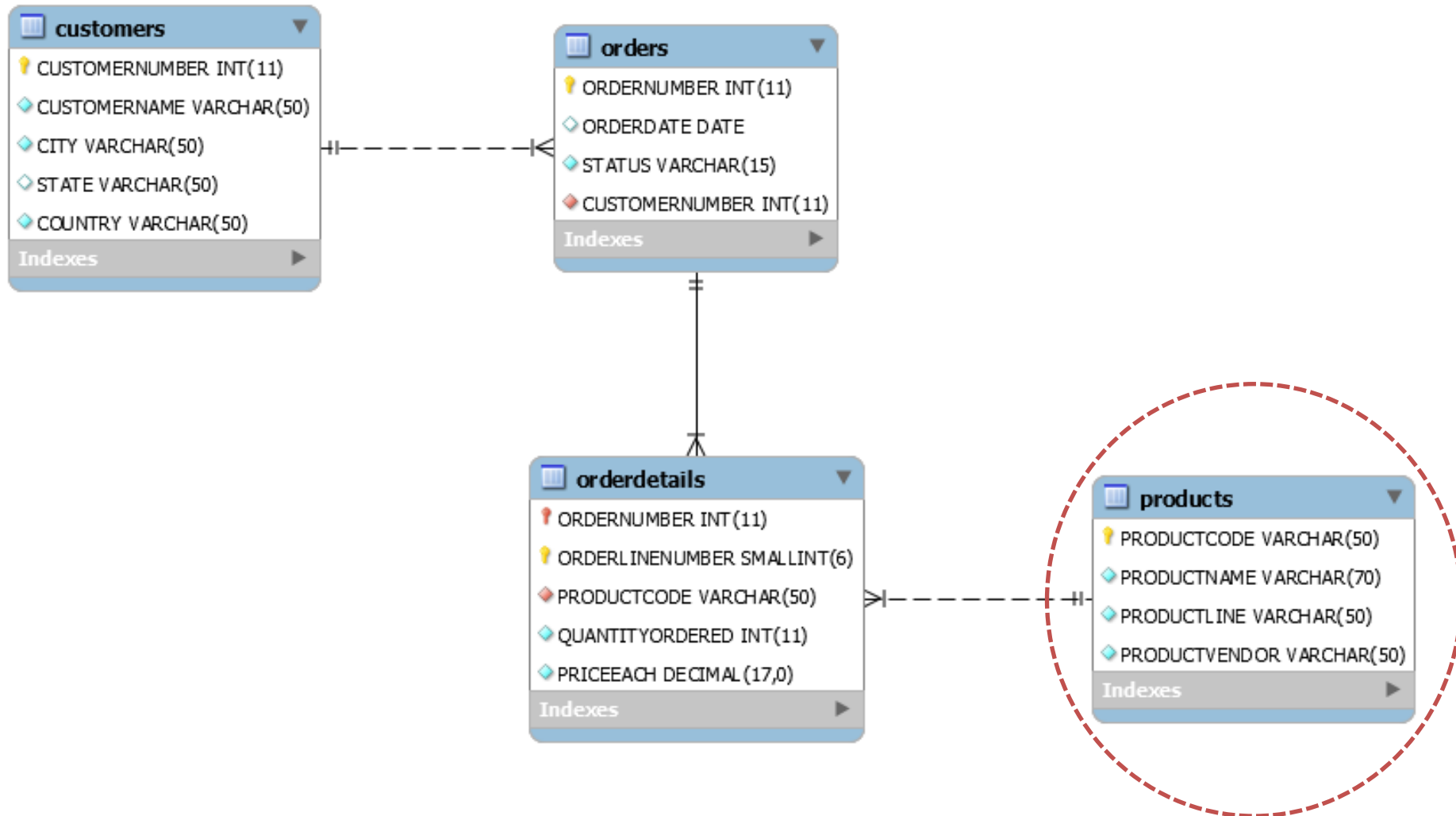
# The products table
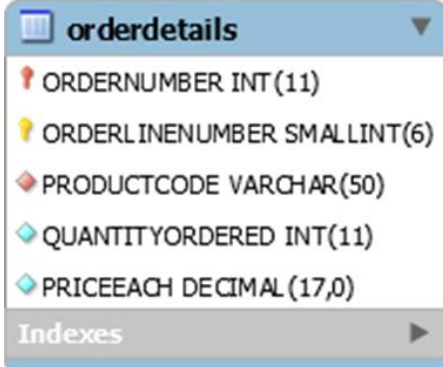
```
+-------------+----------------------------------------+-----------------+-----------------+
| PRODUCTCODE | PRODUCTNAME                            | PRODUCTLINE     | PRODUCTVENDOR   |
+-------------+----------------------------------------+-----------------+-----------------+
| S10_1678    | 1969 Harley Davidson Ultimate Chopper  | Motorcycles     | Min Lin         |
| S10_1949    | 1952 Alpine Renault 1300               | Classic Cars    | Classic Metal   |
| S10_2016    | 1996 Moto Guzzi 1100i                  | Motorcycles     | Highway 66      |
| S10_4698    | 2003 Harley-Davidson Eagle Drag Bike   | Motorcycles     | Red Start       |
| S10_4757    | 1972 Alfa Romeo GTA                    | Classic Cars    | Motor City      |
| S10_4962    | 1962 LanciaA Delta 16V                 | Classic Cars    | Second Gear     |
| S12_1099    | 1968 Ford Mustang                      | Classic Cars    | Autoart Studio  |
| S12_1108    | 2001 Ferrari Enzo                      | Classic Cars    | Second Gear     |
| S12_1666    | 1958 Setra Bus                         | Trucks and Buses| Welly Diecast   |
| S12_2823    | 2002 Suzuki XREO                       | Motorcycles     | Unimax Art      |
| S12_3148    | 1969 Corvair Monza                     | Classic Cars    | Welly Diecast   |
| S12_3380    | 1968 Dodge Charger                     | Classic Cars    | Welly Diecast   |
| S12_3891    | 1969 Ford Falcon                       | Classic Cars    | Second Gear     |
| S12_3990    | 1970 Plymouth Hemi Cuda                | Classic Cars    | Studio M        |
| S12_4473    | 1957 Chevy Pickup                      | Trucks and Buses| Exoto Designs   |
| S12_4675    | 1969 Dodge Charger                     | Classic Cars    | Welly Diecast   |
| S18_1097    | 1940 Ford Pickup Truck                 | Trucks and Buses| Studio M        |
| S18_1129    | 1993 Mazda RX-7                        | Classic Cars    | Highway 66      |
| S18_1342    | 1937 Lincoln Berline                   | Vintage Cars    | Motor City      |
| S18_1367    | 1936 Mercedes-Benz 500K Special Roadster | Vintage Cars  | Studio M        |
+-------------+----------------------------------------+-----------------+-----------------+
```

# Transactional vs. analytical processing

- Transactional processing (OLTP)
  - create a new order
  - add products to an order
  - change the status of an order
  - etc.

- Analytical processing (OLAP)
  - sales by customer country
  - sales by product line
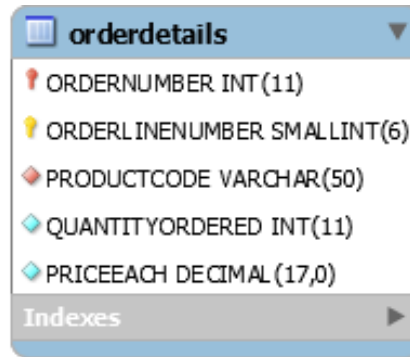  - sales by year
  - etc.

# Analytical queries

- Calculating sales
  - table: orderdetails
  - column: quantityordered
  - column: priceeach
  - multiply: quantityordered*priceeach
  - sum: sum(quantityordered*priceeach)

# Total sales



**select sum**(a.quantityordered*a.priceeach) **as** sales
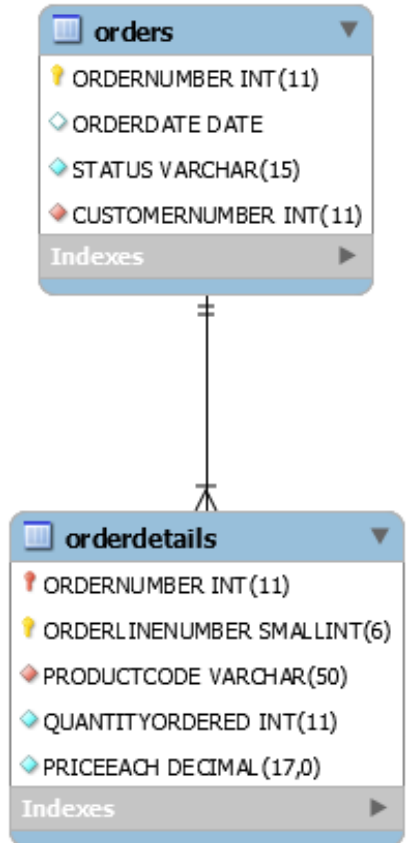**from** orderdetails **as** a;

```
+----------+
| sales    |
+----------+
| 11581065 |
+----------+
1 row in set (0.01 sec)
```

# Sales by customer

**select** b.customernumber,

        **sum**(a.quantityordered*a.priceeach) **as** sales

**from** orderdetails **as** a,

       orders **as** b

**where** a.ordernumber = b.ordernumber

**group by** b.customernumber;

```
+----------------+--------+
| customernumber | sales  |
+----------------+--------+
|             97 | 300000 |
|             98 | 300000 |
|             99 |  30000 |
|            100 |   5160 |
|            103 |  24160 |
|            ... |    ... |
+----------------+--------+
102 rows in set (0.01 sec)
```

orders
- ORDERNUMBER INT(11)
- ORDERDATE DATE
- STATUS VARCHAR(15)
- CUSTOMERNUMBER INT(11)
- Indexes

orderdetails
- ORDERNUMBER INT(11)
- ORDERLINENUMBER SMALLINT(6)
- PRODUCTCODE VARCHAR(50)
- QUANTITYORDERED INT(11)
- PRICEEACH DECIMAL(17,0)
- Indexes

# Sales by customer country
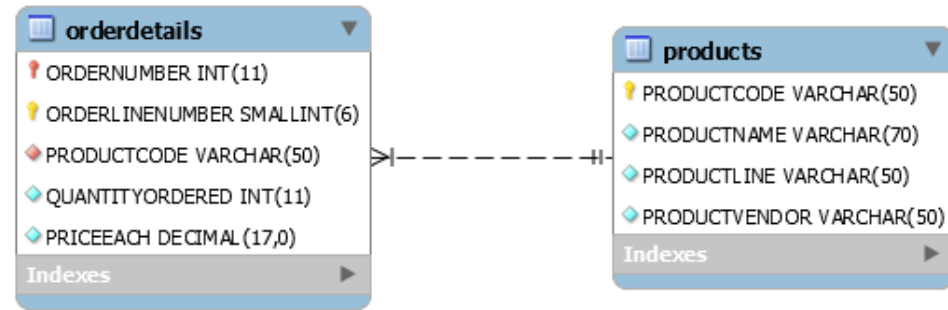


```
select c.country,
        sum(a.quantityordered*a.priceeach) as sales
from orderdetails as a,
        orders as b,
        customers as c
where a.ordernumber = b.ordernumber
   and b.customernumber = c.customernumber
group by c.country;
```

```
+-------------+---------+
| country     | sales   |
+-------------+---------+
| Australia   |  630638 |
| Austria     |  202089 |
| Belgium     |  108485 |
| Canada      |  224085 |
| ...         |     ... |
+-------------+---------+
21 rows in set (0.02 sec)
```

# Sales by product line



```sql
select b.productline,
       sum(a.quantityordered*a.priceeach) as sales
from orderdetails as a,
     products as b
where a.productcode = b.productcode
group by b.productline;
```

```
+------------------+---------+
| productline      | sales   |
+------------------+---------+
| Classic Cars     | 4090489 |
| Motorcycles      | 1274351 |
| Planes           | 1077130 |
| Ships            |  748369 |
| Trains           |  234662 |
| Trucks and Buses | 1154450 |
| Vintage Cars     | 3001614 |
+------------------+---------+
7 rows in set (0.02 sec)
```

# Sales by year

**select year**(b.orderdate) **as** year,
  **sum**(a.quantityordered*a.priceeach) **as** sales
**from** orderdetails **as** a,
  orders **as** b
**where** a.ordernumber = b.ordernumber
**group by year**(b.orderdate);

```
+------+---------+
| year | sales   |
+------+---------+
| 2003 | 4312435 |
| 2004 | 4987780 |
| 2005 | 1980850 |
+------+---------+
3 rows in set (0.02 sec)
```

**orders**
- ORDERNUMBER INT(11)
- ORDERDATE DATE
- STATUS VARCHAR(15)
- CUSTOMERNUMBER INT(11)
- Indexes

**orderdetails**
- ORDERNUMBER INT(11)
- ORDERLINENUMBER SMALLINT(6)
- PRODUCTCODE VARCHAR(50)
- QUANTITYORDERED INT(11)
- PRICEEACH DECIMAL(17,0)
- Indexes
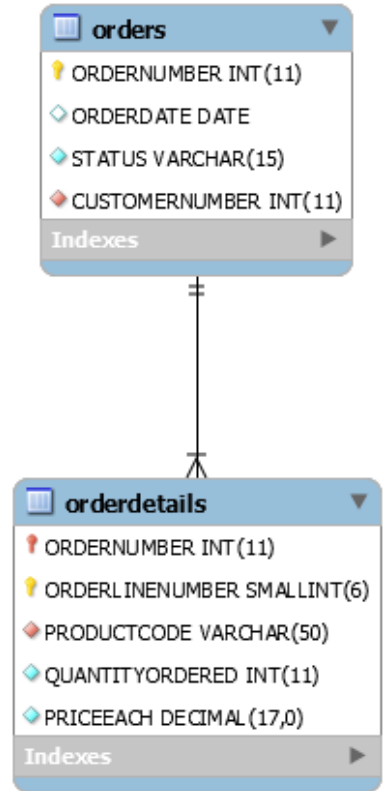
# Multi-dimensional model

- Data can be analyzed according to different **dimensions**
  - in this example
    - customer country
    - product line
    - year
  - what about combining multiple dimensions?

# Sales by product line and year

```
select c.productline,
        year(b.orderdate) as year,
        sum(a.quantityordered*a.priceeach) as sales
from orderdetails as a,
        orders as b,
        products as c
where a.ordernumber = b.ordernumber
    and a.productcode = c.productcode
group by c.productline, year(b.orderdate);
```
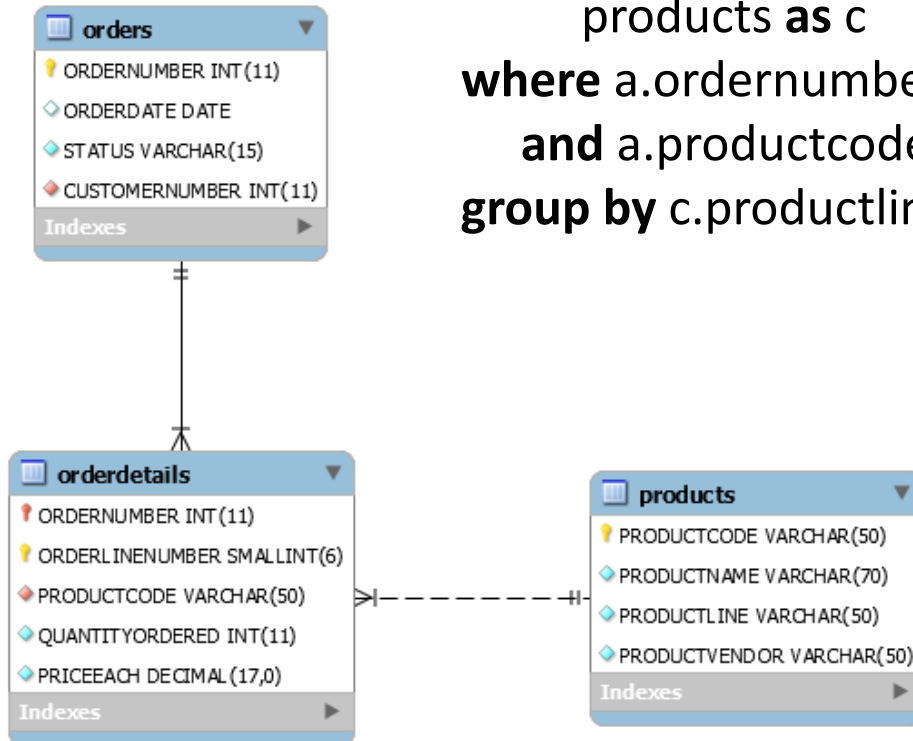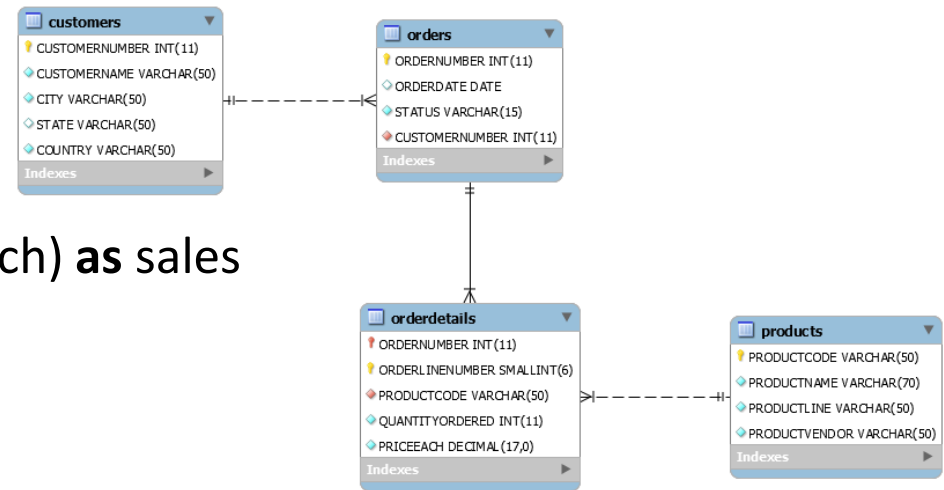
```
+------------------+------+---------+
| productline      | year | sales   |
+------------------+------+---------+
| Classic Cars     | 2003 | 1513998 |
| Classic Cars     | 2004 | 1837904 |
| Classic Cars     | 2005 |  738587 |
| Motorcycles      | 2003 |  397392 |
| Motorcycles      | 2004 |  590632 |
| ...              |  ... |     ... |
+------------------+------+---------+
21 rows in set (0.03 sec)
```

# Sales by customer country, product line and year

**select** d.country,
        c.productline,
        **year**(b.orderdate) **as** year,
        **sum**(a.quantityordered*a.priceeach) **as** sales
**from** orderdetails **as** a,
      orders **as** b,
      products **as** c,
      customers **as** d
**where** a.ordernumber = b.ordernumber
   **and** a.productcode = c.productcode
   **and** b.customernumber = d.customernumber
**group by** d.country,
        c.productline,
        **year**(b.orderdate);

**customers**
- CUSTOMERNUMBER INT(11)
- CUSTOMERNAME VARCHAR(50)
- CITY VARCHAR(50)
- STATE VARCHAR(50)
- COUNTRY VARCHAR(50)
- Indexes

**orders**
- ORDERNUMBER INT(11)
- ORDERDATE DATE
- STATUS VARCHAR(15)
- CUSTOMERNUMBER INT(11)
- Indexes

**orderdetails**
- ORDERNUMBER INT(11)
- ORDERLINENUMBER SMALLINT(6)
- PRODUCTCODE VARCHAR(50)
- QUANTITYORDERED INT(11)
- PRICEEACH DECIMAL(17,0)
- Indexes

**products**
- PRODUCTCODE VARCHAR(50)
- PRODUCTNAME VARCHAR(70)
- PRODUCTLINE VARCHAR(50)
- PRODUCTVENDOR VARCHAR(50)
- Indexes

```
+-------------+------------------+------+--------+
| country     | productline      | year | sales  |
+-------------+------------------+------+--------+
| Australia   | Classic Cars     | 2003 |  85355 |
| Australia   | Classic Cars     | 2004 |  76198 |
| Australia   | Classic Cars     | 2005 |  31460 |
| Australia   | Motorcycles      | 2003 |  42337 |
| Australia   | Motorcycles      | 2004 |  33077 |
| ...         | ...              | ...  |   ...  |
+-------------+------------------+------+--------+
245 rows in set (0.04 sec)
```

# Multi-dimensional model

- Analytical queries over the database
  - non-uniform treatment of dimensions
    - e.g. use of special functions for the time dimension
  - complex aggregates to calculate measures
    - e.g. sum(quantityordered*priceeach)
  - some calculations are done over and over again
    - e.g. quantityordered*priceeach
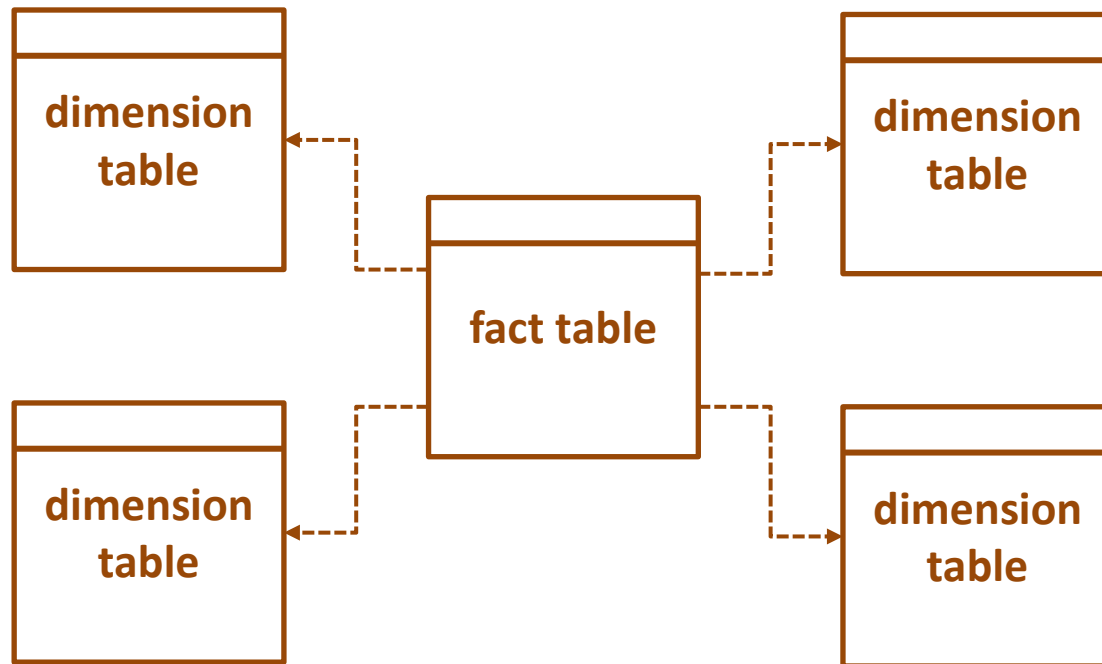  - all of this running in parallel with transactional queries
    - performance impact

# Data warehousing

- Advantages of having a separate data warehouse
  - separate OLTP and OLAP operations
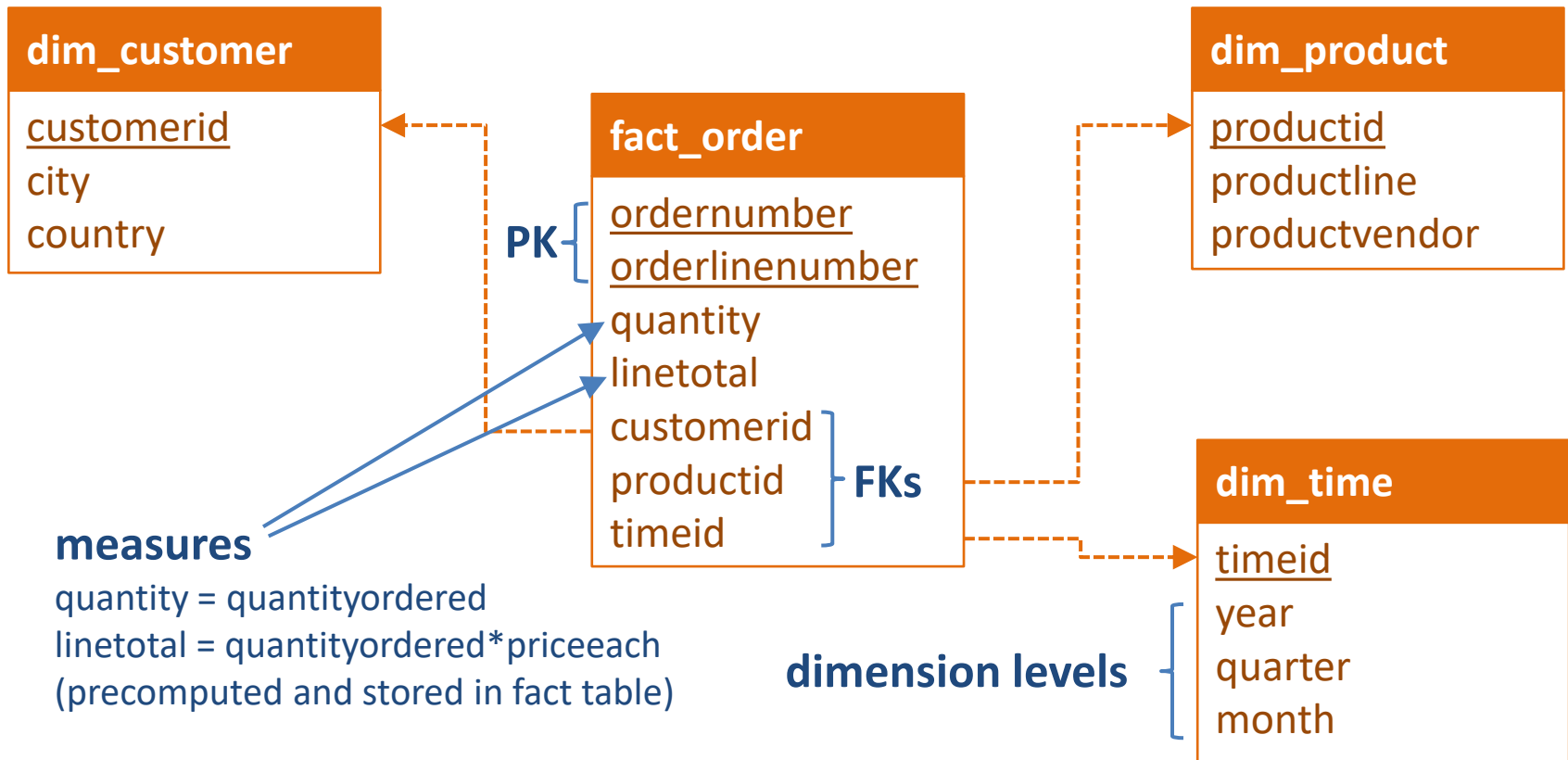  - design a suitable schema for analytical queries
    - star schema
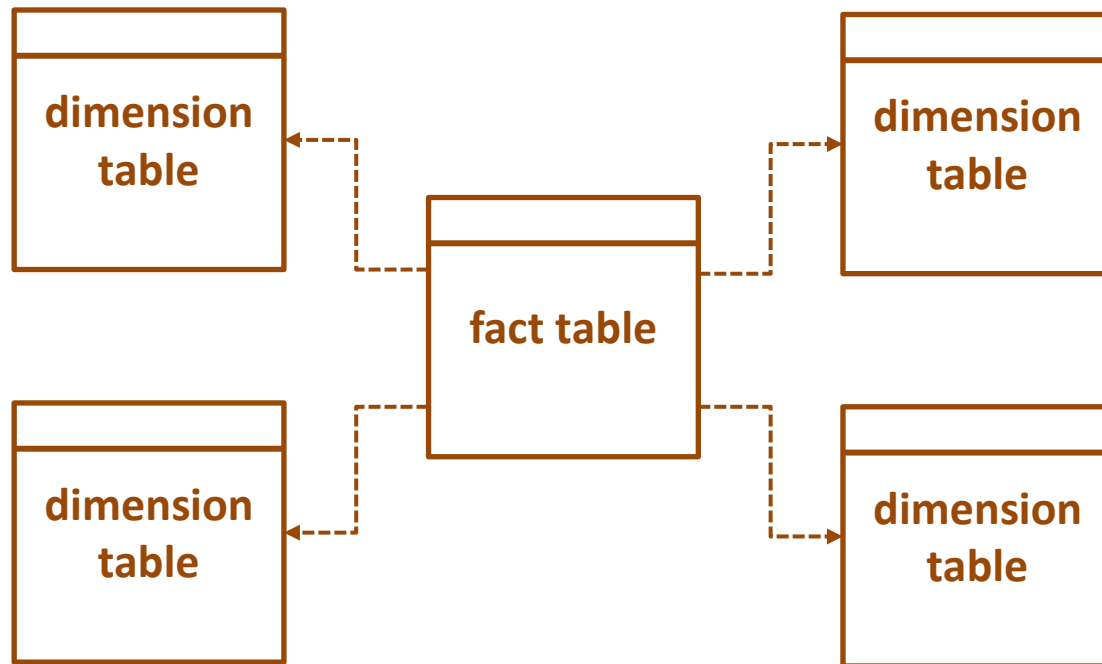
# Data warehousing

- Analytical queries over a star schema
  - join fact table with the desired dimension tables
  - group by some level of those dimensions
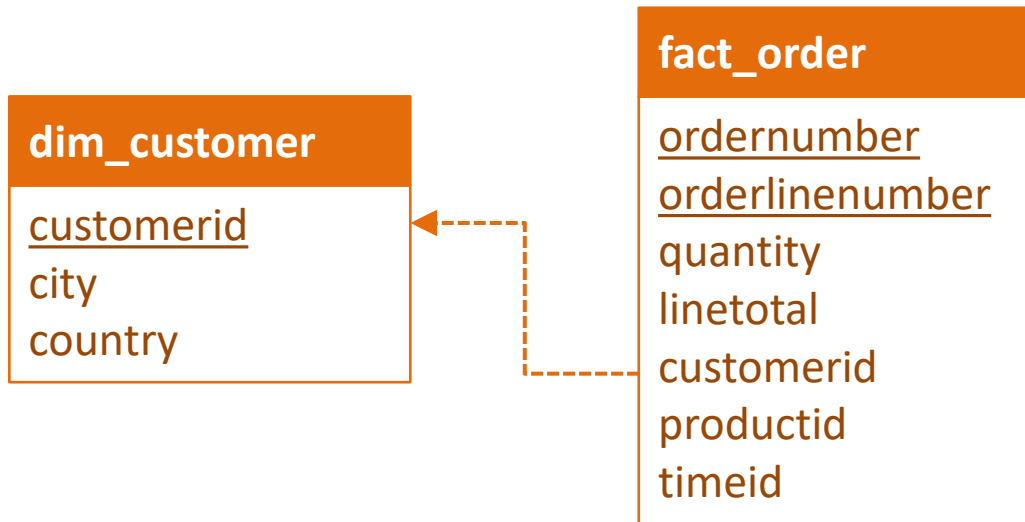  - aggregate some measure in the fact table

# Data warehousing

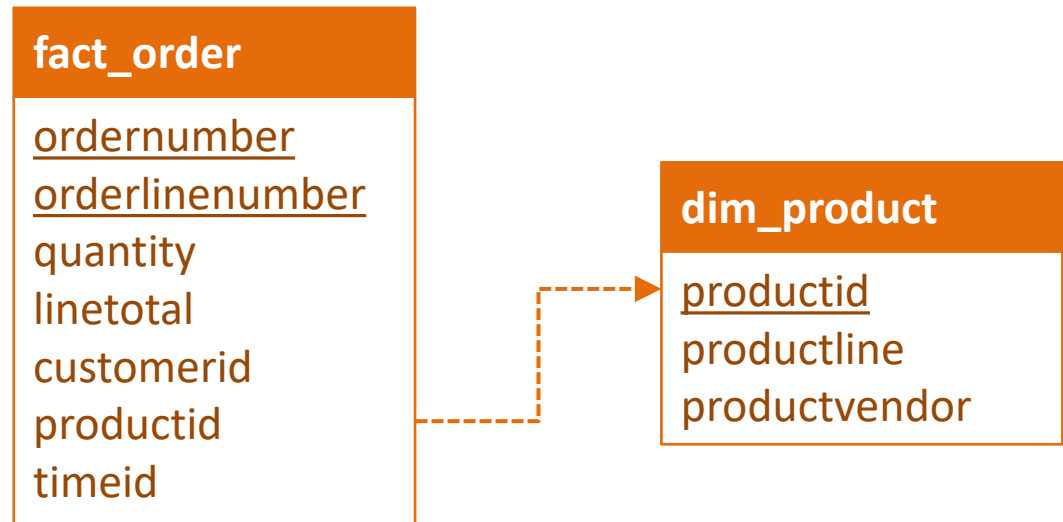- Example of a star schema



**dim_customer**

customerid
city
country

**fact_order**

PK {
ordernumber
orderlinenumber
}
quantity
linetotal
customerid
productid
timeid
} FKs

**dim_product**

productid
productline
productvendor

**dim_time**

timeid
year
quarter
month
} dimension levels

**measures**
quantity = quantityordered
linetotal = quantityordered*priceeach
(precomputed and stored in fact table)

# Data warehousing

- Sales by customer country

  **select** b.country, **sum**(a.linetotal) **as** sales
  **from** fact_order **as** a, dim_customer **as** b
  **where** a.customerid = b.customerid
  **group by** b.country;

**dim_customer**

customerid
city
country

**fact_order**

ordernumber
orderlinenumber
quantity
linetotal
customerid
productid
timeid

# Data warehousing

- Sales by product line

**select** b.productline, **sum**(a.linetotal) **as** sales
**from** fact_order **as** a, dim_product **as** b
**where** a.productid = b.productid
**group by** b.productline;

**fact_order**

ordernumber
orderlinenumber
quantity
linetotal
customerid
productid
timeid

**dim_product**

productid
productline
productvendor

# Data warehousing

- Sales by year

**select** b.year, **sum**(a.linetotal) **as** sales
**from** fact_order **as** a, dim_time **as** b
**where** a.timeid = b.timeid
**group by** b.year;

**fact_order**

ordernumber
orderlinenumber
quantity
linetotal
customerid
productid
timeid

**dim_time**

timeid
year
quarter
month
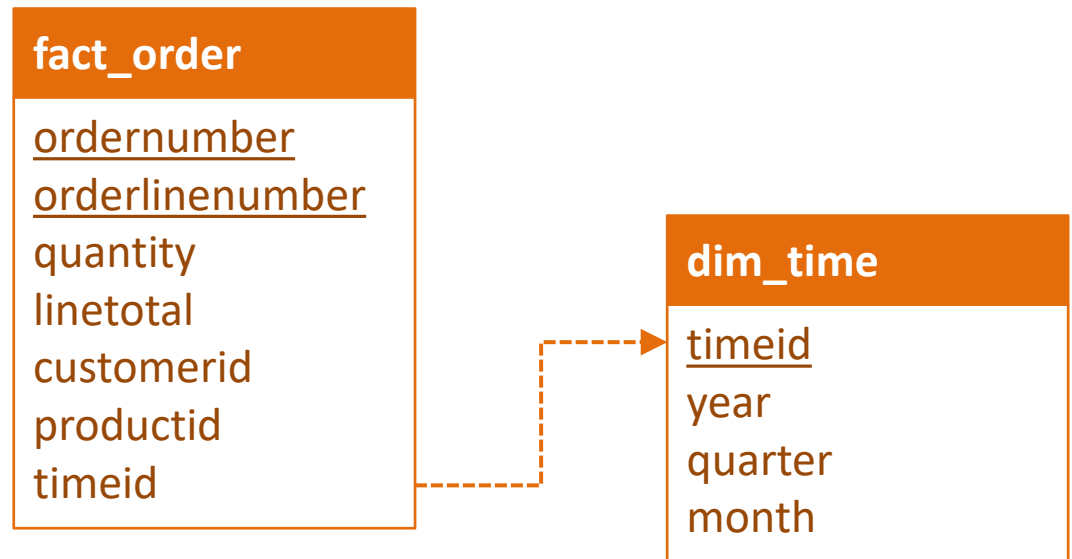
# Data warehousing

- Sales by customer country and product line

**select** c.country, b.productline, **sum**(a.linetotal) **as** sales
**from** fact_order **as** a, dim_product **as** b, dim_customer **as** c
**where** a.productid = b.productid **and** a.customerid = c.customerid
**group by** c.country, b.productline;

**dim_customer**

customerid
city
country

**fact_order**

ordernumber
orderlinenumber
quantity
linetotal
customerid
productid
timeid

**dim_product**

productid
productline
productvendor
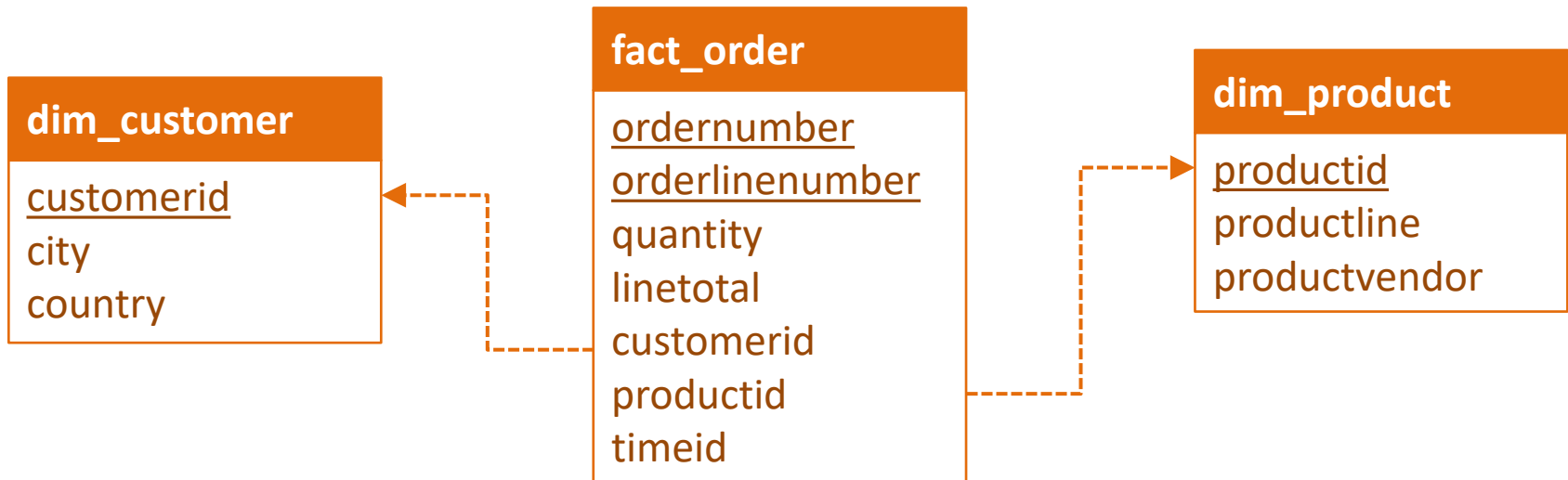
# Data warehousing

- Sales by customer country and year

   **select** c.country, b.year, **sum**(a.linetotal) **as** sales
   **from** fact_order **as** a, dim_time **as** b, dim_customer **as** c
   **where** a.timeid = b.timeid **and** a.customerid = c.customerid
   **group by** c.country, b.year;
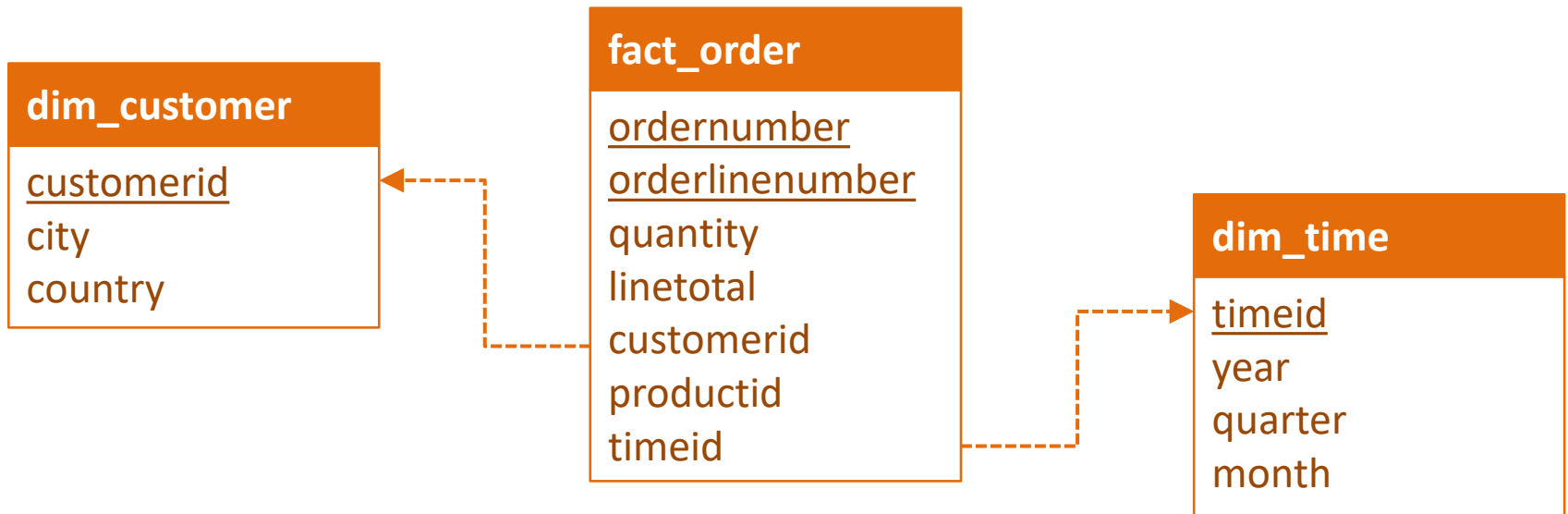
# Data warehousing

- Sales by product line and year
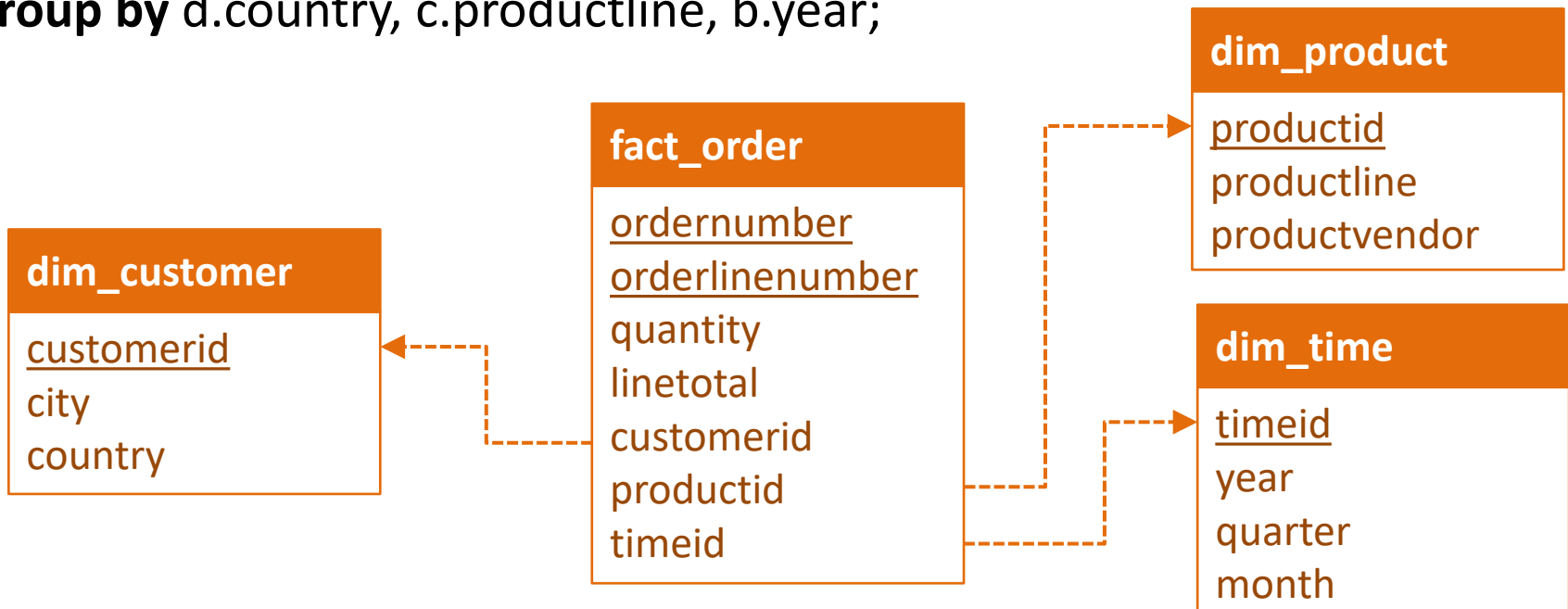
  **select** c.productline, b.year, **sum**(a.linetotal) **as** sales
  **from** fact_order **as** a, dim_time **as** b, dim_product **as** c
  **where** a.timeid = b.timeid **and** a.productid = c.productid
  **group by** c.productline, b.year;
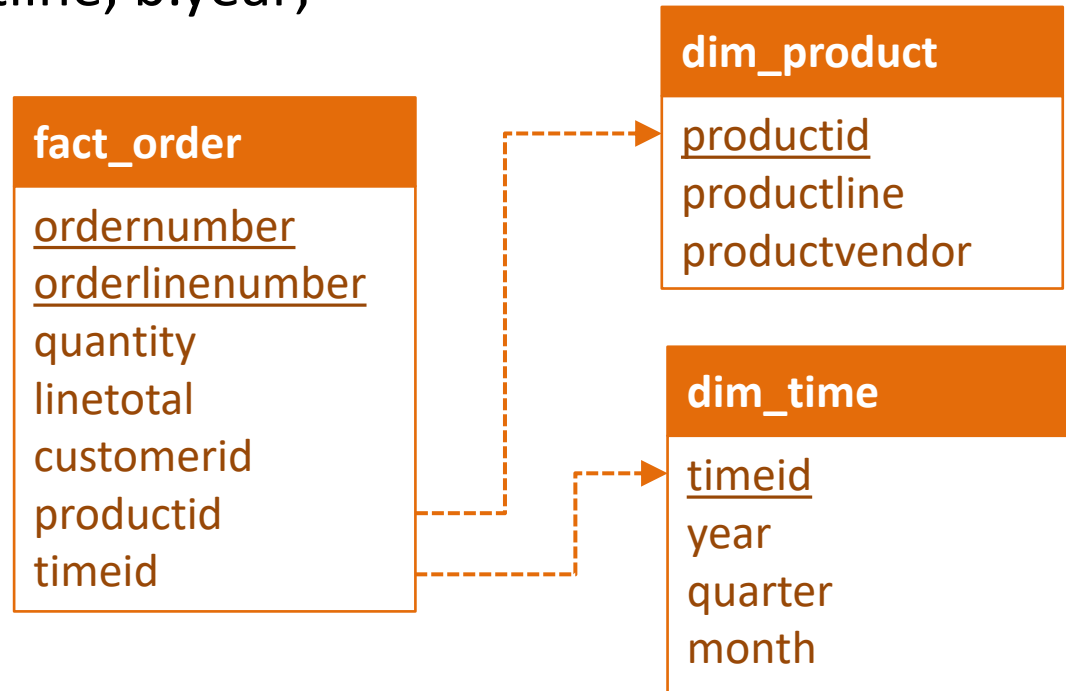


**dim_product**

productid
productline
productvendor

**fact_order**

ordernumber
orderlinenumber
quantity
linetotal
customerid
productid
timeid

**dim_time**

timeid
year
quarter
month

# Data warehousing

- Sales by customer country, product line and year

**select** d.country, c.productline, b.year, **sum**(a.linetotal) **as** sales
**from** fact_order **as** a, dim_time **as** b, dim_product **as** c, dim_customer **as** d
**where** a.timeid = b.timeid **and** a.productid = c.productid **and**
      a.customerid = d.customerid
**group by** d.country, c.productline, b.year;

**dim_product**

productid
productline
productvendor

**fact_order**

ordernumber
orderlinenumber
quantity
linetotal
customerid
productid
timeid

**dim_customer**

customerid
city
country

**dim_time**

timeid
year
quarter
month

# Data warehousing

- ## How to build a data warehouse
  - ### ETL process
    - extract data from original database
    - transform data to fit star schema
    - load data onto data warehouse



database

data warehouse

ETL
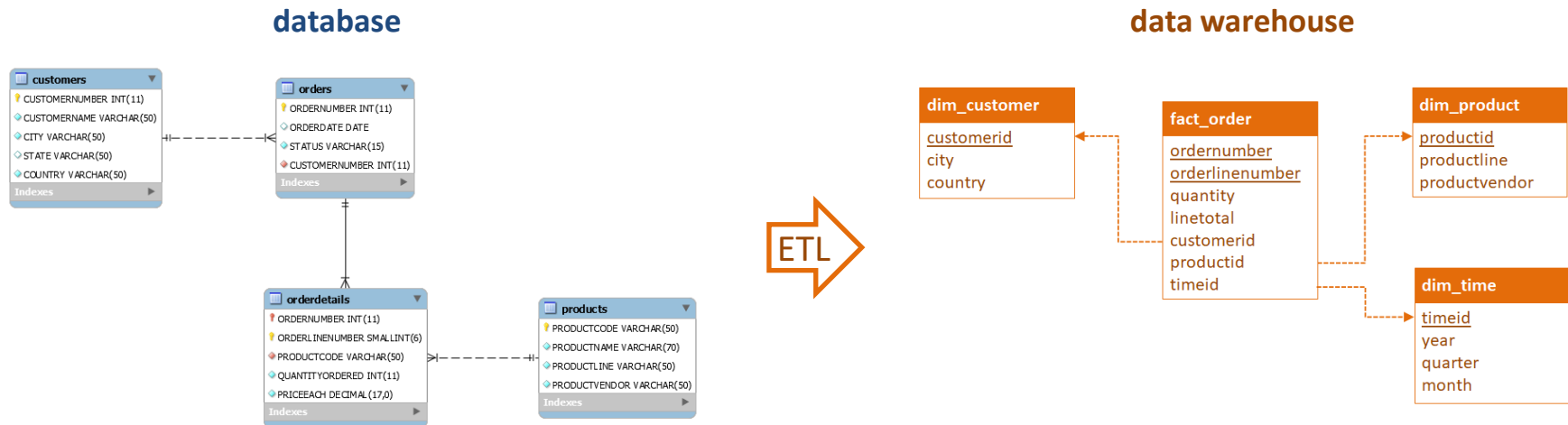
# Data warehousing

- How to build a data warehouse
  - this usually involves
    - one transformation for each dimension table
    - one transformation for the fact table
    - a job that runs all transformations in the correct sequence