1. SELECT запрос:

Запрос для отображения текущих цен на товары в заказе:

```
SELECT ord.orderid, dtl.orderline ,
        p.name as product,
        v.name as vendor,
        pv.unitcost as price
FROM (((orders.order as ord INNER JOIN
        orders.orderdtl as dtl ON
        ord.ordernumber= '12345A' AND
        ord.OrderId = dtl.orderid) INNER JOIN
        logistic.product as p ON
        p.productid = dtl.productid) LEFT JOIN
        logistic.vendor as v ON
        v.vendorid = dtl.vendorid) LEFT JOIN
        logistic.productvendor as pv ON
        pv.vendorid = dtl.vendorid and
        pv.productid = dtl.productid;
```

```
myshop=# select ord.orderid, dtl.orderline ,
myshop-#        p.name as product,
myshop-#        v.name as vendor,
myshop-#        pv.unitcost as price
myshop-# from (((orders.order as ord inner join
myshop(#        orders.orderdtl as dtl on
myshop(#        ord.ordernumber = '12345A' and
myshop(#        ord.OrderId = dtl.orderid) inner join
myshop(#        logistic.product as p on
myshop(#        p.productid = dtl.productid) left join
myshop(#        logistic.vendor as v on
myshop(#        v.vendorid = dtl.vendorid) left join
myshop-#        logistic.productvendor as pv on
myshop-#        pv.vendorid = dtl.vendorid and
myshop-#        pv.productid = dtl.productid;
 orderid | orderline |    product    |     vendor     |  price
---------+-----------+---------------+----------------+---------
       4 |         1 | Брюки синие   | ООО Баттон Блю | 1100.00
       4 |         2 | Брюки зеленые | ООО Баттон Блю | 1500.00
(2 rows)
```

2. Напишите запрос по своей базе с использованием LEFT JOIN и INNER JOIN, как порядок соединений в FROM влияет на результат? Почему?

Планировщик может соединять данные таблицы в любом порядке , как ему покажется более эффективным исполение, поэтому для однозначного выполнения запроса лучше воспользоваться скобками.
Например,
SELECT * FROM a LEFT JOIN b ON (a.bid = b.id) LEFT JOIN c ON (a.cid = c.id);
можно соединить A либо с B, либо с C.

Но этот пример на моих данных поймать мне не удалось.

А вот если из условия ON вынести проверку в WHERE,
то результаты получаются очевидно разные, потому как условие WHERE
применяется уже в самом конце к получившемуся результату:

```
-- этот запрос возвращает 4 сточки - одна из которых содержит NULL в поле price
SELECT dtl.orderid, dtl.orderline, b.unitcost as price, c.name as product
FROM orders.orderdtl as dtl
LEFT JOIN logistic.productvendor as b ON b.vendorid = dtl.vendorid and b.productid = dtl.productid
LEFT JOIN logistic.product as c ON c.productid = dtl.productid
```

```
myshop=# SELECT dtl.orderid, dtl.orderline, b.unitcost as price, c.name as product
myshop-# FROM orders.orderdtl as dtl
myshop-# LEFT JOIN logistic.productvendor as b ON b.vendorid = dtl.vendorid and b.productid = dtl.productid
myshop-# LEFT JOIN logistic.product as c ON c.productid = dtl.productid;
 orderid | orderline |  price  |    product
---------+-----------+---------+---------------
       4 |         1 | 1100.00 | Брюки синие
       4 |         2 | 1500.00 | Брюки зеленые
       5 |         1 | 7200.00 | Куртка
       5 |         2 |         | Шапка
(4 rows)
```

```
-- а этот только 3
SELECT dtl.orderid, dtl.orderline, b.unitcost as price, c.name as product
FROM orders.orderdtl as dtl
LEFT JOIN logistic.productvendor as b ON b.productid = dtl.productid
LEFT JOIN logistic.product as c ON c.productid = dtl.productid
WHERE b.vendorid = dtl.vendorid
```

```
myshop=# SELECT dtl.orderid, dtl.orderline, b.unitcost as price, c.name as product
myshop-# FROM orders.orderdtl as dtl
myshop-# LEFT JOIN logistic.productvendor as b ON b.productid = dtl.productid
myshop-# LEFT JOIN logistic.product as c ON c.productid = dtl.productid
myshop-# where b.vendorid = dtl.vendorid
myshop-# ;
 orderid | orderline |  price  |    product
---------+-----------+---------+---------------
       4 |         1 | 1100.00 | Брюки синие
       4 |         2 | 1500.00 | Брюки зеленые
       5 |         1 | 7200.00 | Куртка
(3 rows)
```

3. Запрос на добавление данных с выводом информации о добавленных строках.

```
INSERT INTO logistic.product(
        productid, name, description, age, size)
VALUES (DEFAULT, 'Artie Шорты для девочки', '97% хлопок, 3% эластан', 3, 98)
RETURNING (productid, name);
```

```
myshop=# INSERT INTO logistic.product(
myshop(# productid, name, description, age, size)
myshop-# VALUES (DEFAULT, 'Artie Шорты для девочки', '97% хлопок, 3% эластан', 3, 98)
myshop-#     RETURNING (productid, name);
              row
--------------------------------
 (17,"Artie Шорты для девочки")
(1 row)
```

4. Запрос с обновлением данные используя UPDATE FROM

```
-- нужно пересчитать цены в заказе
UPDATE orders.orderdtl AS dtl
SET unitcost = pv.unitcost,
  price = pv.unitcost - dtl.discount
FROM logistic.productvendor as pv, orders.order as o
WHERE pv.vendorid = dtl.vendorid and
 pv.productid = dtl.productid and
 o.orderid = dtl.orderid and
 o.ordernumber = '12345A'
RETURNING (dtl.orderid, dtl.orderline, dtl.productid, dtl.unitcost, dtl.discount, dtl.price);
```

```
myshop=#
myshop=# UPDATE orders.orderdtl AS dtl
myshop-# SET unitcost = pv.unitcost,
myshop-#     price = pv.unitcost - dtl.discount
myshop-# FROM logistic.productvendor as pv, orders.order as o
myshop-# WHERE pv.vendorid = dtl.vendorid and
myshop-#    pv.productid = dtl.productid and
myshop-#    o.orderid = dtl.orderid and
myshop-#    o.ordernumber = '12345A'
myshop-# RETURNING (dtl.orderid, dtl.orderline, dtl.productid, dtl.unitcost,
myshop(#    dtl.discount, dtl.price);
             row
-------------------------------
 (4,1,12,1100.00,0.00,1100.00)
 (4,2,13,1500.00,0.00,1500.00)
(2 rows)
```

5. Запрос для удаления данных с оператором DELETE используя join с другой таблицей с помощью using.

```
DELETE FROM orders.orderdtl USING logistic.product as p
WHERE orders.orderdtl.productid = p.productid AND
p.Name = 'Шапка'
RETURNING (orderid, orderline, orders.orderdtl.productid);
```

```
myshop=# DELETE FROM orders.orderdtl AS dtl USING logistic.product as p
myshop-# WHERE dtl.productid = p.productid AND p.Name = 'Шапка'
myshop-# RETURNING (dtl.orderid, dtl.orderline, dtl.productid);
   row
----------
 (5,2,16)
(1 row)

DELETE 1
myshop=#
```

6. Пример использования утилиты COPY

Утилиту COPY можно использовать для выгрузки данных и таблицы (COPY TO) в файл и заливки данных обратно (COPY FROM)

```
COPY (SELECT * FROM logistic.productvendor)
TO '/var/run/archive-dir/copytest.csv'
(DELIMITER ';');

delete from logistic.productvendor;

COPY logistic.productvendor
FROM '/var/run/archive-dir/copytest.csv'
(DELIMITER ';');
```

```
myshop=# select * from logistic.productvendor;
 vendorid | productid | unitcost
----------+-----------+----------
        8 |        14 |  1200.00
        9 |        12 |  1100.00
        9 |        13 |  1500.00
       10 |        15 |  7200.00
(4 rows)

myshop=# COPY (SELECT * FROM logistic.productvendor)
myshop-# TO '/var/run/archive-dir/copytest.csv'
myshop-# (DELIMITER ';');
COPY 4
myshop=# delete from logistic.productvendor;
DELETE 4
myshop=# select * from logistic.productvendor;
 vendorid | productid | unitcost
----------+-----------+----------
(0 rows)

myshop=# COPY logistic.productvendor
myshop-# FROM '/var/run/archive-dir/copytest.csv'
myshop-# (DELIMITER ';');
COPY 4
myshop=# select * from logistic.productvendor;
 vendorid | productid | unitcost
----------+-----------+----------
        8 |        14 |  1200.00
        9 |        12 |  1100.00
        9 |        13 |  1500.00
       10 |        15 |  7200.00
(4 rows)

myshop=#
```