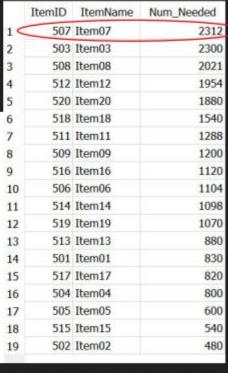
SQLite

DB Browser for SQLite Version 3.12.1 SQLite database with 21 tables

Which component is most needed across products? Order the components required in descending order.

```
SELECT
2
         VersionItems.ItemID,
3
         Component. ItemName,
         sum (Product. InitQuantity *VersionItems. Quantity) as Num Needed
5
     FROM
6
         Product join Design on Product.ProductID = Design.ProductID
7
         join VersionItems on Design. VersionID = VersionItems. VersionID
8
         join Component on Component. ItemID = VersionItems. ItemID
9
     Group by
10
         Component.ItemID
11
     HAVING
12
         Design.Date = max (Design.Date)
13
     Order by
         Num Needed DESC
```



In order to find the most needed component across products, we performed a Groupby on the Component's ItemID. For this to make sense, we had to join several tables. We joined our way from Product to Component tables, through Design and VersionItems. A product has many versions (VersionID), and each version has a list of items (listed in VersionItems) with its own quantity listed. Product versions are sequential (the most recently created is an update of the previous), so we only selected those versions where the Design. Date was the maximum. For each Product, the Customer specified an Initial Quantity that they would request. If you multiply this by the quantity of each Item, you get the total number of required components (of a particular type) for an entire order. To do this across all products, we did a sumproduct of InitQuantity * VersionItemsQuantity. Finally, we ordered the results in descending order. The most needed component is Item07.

Which supplier has the lowest quality? Use overall score average.

To find the average score of a Supplier's Items, we joined Supplier to Component to Test (where the OverallScore resides). Performing a Groupby on Supplier, we averaged each supplier's OverallScore (for all items they sold), and ordered by Ascending. Supplier01 had the lowest quality.

1	SELECT			
2	Suppli	Supplier.SupplierName, AVG(Test.OverallScore) as Average_Score FROM		
	AVG(7			
4				
5		CARLO STATE OF THE	ent on Component.SupplierID = Supplier.SupplierID	
6			nt.ItemID = Test.ItemID	
		ier.SupplierID		
9		75/75/2017		
10	Test.O	verallScore ASC		
	SupplierName	Average_Score		
<	Supplier01	1.5		
2	Supplier02	3.5		
3	Supplier03	5.5		
4	Supplier04	7.5		
5	Supplier05	9.5		
6	Supplier06	11.5		
7	Supplier07	14.5		
	Supplier08	15.5		
8		16.5		
8	Supplier09	10.5		

Identify employees that are not assigned to a production stage.

```
1    SELECT
2     Employee.EmployeeName
3     FROM
4     Employee
5     EXCEPT
6     SELECT
7     Employee.EmployeeName
8     FROM
9     Employee join Station on Employee.EmployeeID = Station.EmployeeID;
```

Since an employee cannot be assigned to a production stage (i.e. have a StageNumber) without being on a Station, we are looking for employees whose EmployeeID's are not in the Station table. To solve this problem, all we needed was a simple set minus. Using EXCEPT, we subtracted the set of all employees from the set of employees with a production stage. The query takes EmployeeID's from the Employee table and EXCEPTs EmployeeID's of the join of the Employee table and the Station table. In total, there are 29 employees who are not assigned to a production stage, out of 39 employees. 10 of those are supervisors (which oversee shifts, not production stages), 1 is a janitor, and the other 9 are unassigned workers.

	EmployeeName
	- Control of the state of the s
	Chad
6	Charles
7	Dana
8	Debora
9	Don
10	Doris
11	Felisha
12	Fred
13	Greg
14	Hubber
15	Jacqueline
16	James
17	John
18	Link
19	Moore
20	Olan
21	Quincy
22	Rhett
23	Rick
24	Rutherford
25	Ryu
26	Tristan
27	Vivian
28	Xia
29	Zane

Who is the biggest customer by item in terms of shipped product quantity?

```
SELECT
                                                                                            ShippedProductQuantity
                                                                        CompanyName
   Customer.CompanyName,
   Sum (PacketProducts.Quantity) as ShippedProductQuantity
                                                                      CompanyE
FROM
   Customer join Packet on Customer.CompanyID = Packet.CompanyID
                                                                    2 CompanyD
   join PacketProducts on PacketProducts.PacketID = Packet.PacketID
Group by
                                                                    3 CompanyA
   Customer.CompanyID
                                                                    4 CompanyC
Order by
   ShippedProductQuantity DESC
                                                                    5 CompanyB
```

To find the biggest Customer, we performed a Groupby on CompanyID (we work with companies). Since the question pertains to shipped product quantity, we joined the tables Customer, Packet, and PacketProducts, the latter of which contained the quantity of the particular products we were interested in. We grouped over each CompanyID, summing their Quantities, and ordered by descending. Our biggest customer is CompanyE, at 14 shipped products.