5 2017 SQL Final

This exam was given at the masters level in 2017 for a course just covering SQL. Students had two hours to complete it.

Use the following tables when answering the questions. The tables below consist of information from a manufacturing company's internal database system. This company makes children's toys by assembling *parts* into *items*.

- Columns with the same name can be assumed to be the same.
- The company takes *parts* and turns them into *items* which are then sold.
- **Unless stated above, all values are not null.**
- Parts table:
  - **PID:** The ID for a particular part. (int)
  - **Price:** The cost associated with the part. (float)
  - **Desc:** A short part description. (string)
  - You can assume that PID is unique to each row.
- Item table:
  - **IID:** The Item ID for a particular item. (int)
  - **Desc:** A short item description. (string)
  - **TPlus:** This is a 1 or 0, depending on if the toy is designed for children who are "Twelve Plus" years old. (int)
  - You can assume that IID is unique to each row.
- Assembly Table:
  - This table contains the "recipe" for making each item. It is a map between the parts and the items.
  - **IID:** This is the item ID of the item that is being assembled. (int)
  - **PID:** This is the PID of the part that is going into the item. (int)
  - **NoPart:** This is the quantity of each part that is required to make the item. (int)
- Sales Table:
  - This table contains information regarding the company's sales.
  - **IID:** This is the item ID for the item sold. (int)
  - **Rev:** This is the amount that the item was sold for. (float)
  - **CustID:** This is the ID for the customer. (int)
  - **SalesID:** This is the ID of the salesperson. (int)
  - **DT:** This is the date that the sales took place. (date)
- SP Table:
  - This table contains information regarding the salespeople in the company.

– Each row represents a salesperson.

– **SalesID:** This the ID of the salesperson. (int)

– **ST:** This is the state that the salesperson lives. If a state is NULL, that means the salesperson lives internationally. (string)

– **Name:** This is the salesperson's name.

– **Bonus:** This is equal to "H" or "L" for "High" or "Low" bonus structure. (string)

– You can assume that SalesID is unique to each row.

Table D.1: *Parts* Table, 4,525 Rows

PID	Price	Desc
1	11.99	Plastic Box (11 x 11)
2	12.85	8" Wheel
3	127.85	4" dowel

Table D.2: *Item* Table, 525 Rows

IID	Desc	TPlus
1	Small Wagon	0
2	Children's playhouse	0
3	Plastic Truck	1

Table D.3: *Assembly* Table, 15,225 Rows

IID	PID	NoPart
1	2	4
1	12	1
1	8	2

Table D.4: *Sales* Table, 45,258 Rows

IID	Rev	CustID	SalesID	Dt
12	65.73	1	12	01-03-2011
12	75.73	3	12	02-03-2011
48	265.04	2	17	01-05-2011
92	554.36	85	8	08-27-2011
115	18.18	92	22	08-11-2011

Table D.5: *SP* Table, 35 Rows

SalesID	ST	Name	Bonus
1	CA	Eldon Tyrell	H
2	PA	J.F. Sebastian	L
3		Roy Batty	L
4	CA	Rick Deckard	H

For the following questions write a query that returns the answer **and ONLY the answer.** All questions should be answered with a single SQL query, unless stated otherwise.

1. What are the top five salesperson (SalesID) in terms of *number* of items sold?

```
select SalesID
from sales
group by 1
order by sum(1) desc
limit 5;
```

2. How many salespeople are from California ("CA")?

```
select
        sum(1)
from
        SP
where st = 'CA';
```

3. Write a query which returns three columns and twelve rows. The first column should be month and should be the month that the sales took place, the second column should contain the number of items sold for the "Twelve Plus" audience and the third should contain the number of items sold for the not "Twelve Plus" audience. Assume that the sales table contains all information for the year 2011 (and no other year.

```
select
        date_part('month', dt) as mnth
        , sum( case when TPlus = 1 then 1 else 0 end ) as tplus
        , sum( case when TPlus = 0 then 1 else 0 end ) as tminus
from
        sales
left join
        item
using(iid)
group by 1;
```

4. Write a query which returns 5 columns. The first column is the state, the second column is the number of items sold to that state, the third is the number of items sold to that state which had a revenue greater than $1,000, the fourth column should the number of unique customers sold to that state and the final column should be the number of customers who spent more than $1,000 in a single transaction.

```
select
        lhs.st
        , count(1) as numitemsold
        , sum(case when Rev > 1000 then 1 else 0 end) as numitems1000
        , count(distinct CustID) as unique custs
        , count(distinct case when Rev > 1000 then custID else null end)
from
        SP
left join
        Sales
using( SalesID)
group by 1;
```

5. Write a query which returns IID and the average revenue generated for that item. Exclude items that have been sold less that 10 times.

```
select IID, avg( rev) as avgrev
from sales
group by 1
having count(1) > 10;
```

6. Items which are sold by salepeople in New York ("NY") have to add a tax of 5%. What is the total amount of tax that the company needs to add?

```
select
        .05 * sum( Rev ) as tax
from
        sales
where
        salesid in
                (select salesID from SP where st = 'NY')
```

7. Which item (IID) uses the most unique parts (PID)?

```
select
        IID
from
        Assembly
group by 1
order by count(1) desc
limit 1;
```

8. Which items uses the most *total* parts?

```
select
        IID
from
        Assembly
group by 1
order by sum(NoPart) desc
limit 1;
```

9. Return a list of the items (IID) that have never been sold:

```
select item.IID
from
        item
left join
        sales
using(IID)
where sales.IID is null
```

10. Return IID and the total cost of the parts used to make a one of them.

```
select
        IID, sum( price * noPart) as cost
from
        assembly
left join
        parts
using(PID)
group by IID;
```

11. Of all the salespeople from California ("CA") or Pennsylvania ("PA"), return the one (Name and SalesID) with the highest amount of revenue.

```
select
        SP.SalesID, Name
from
        SP
left join
        Sales
using( SalesID)
where SP.state in ('CA', 'PA')
group by 1,2
order by sum( Rev) desc
limit 1;
```

12. Of all the salespeople (SalesID) who have sold an item for more than $100.00, return the average revenue per item sold on all of their sales.

```
select
        avg( rev) as ar
from
        sales
where
        salesid in
                (select distinct salesid from sales where rev > 100);
```

13. For each bonus structure ("H" or "L") return the percentage of revenue generated from salespeople in that bonus structure. This should return two rows and two columns.

```
select
        bonus,
        sr / sum( sr) over() as pct
from
        (select
                Bonus, sum(rev) as sr
        from
                SP
        left join
                sales
        using( SalesID)
        group by 1) as IQ
```

14. For each Bonus Structure ("H" or "L") return the percentage of revenue generated from salespeople in that bonus structure. This should return one row and two columns (PctRevH and PctRevL). Note that this is the same data as the previous query, but shaped wide, rather than long.

```
select
        sum( case when Bonus = 'H' then rev else 0 end )/sum( Rev) as PctRevH
        , sum( case when Bonus = 'L' then rev else 0 end )/sum( Rev) as PctRevL
from
        SP
left join
        sales
using( SalesID)
```

15. Calculate the total profit (total revenue - total cost) for each item (IID).

```
select
        (SR - ct * totalCost) as profit, lhs.IID
from
        (select sum(NoPart * Price ) as totalCost, IID
                from Assembly left join Part using(PID) group by 2) as lhs
left join
        (select sum( rev ) as SR, IID, count(1) as ct from sales group by 2) as lhs
using( IID )
group by 2;
```

16. Of all the items (IID) which use more than 5 unique PIDs, which two have the largest number of sales (in terms of count)?

```
select
        lhs.IID
from
        (select
                IID
        from
                Assembly
        group by 1
        having count(1) > 5 ) as lhs
left join
        Sales
using(IID)
group by 1
order by count(1) desc
limit 2;
```

17. An item is called "complex" if the ratio of unique parts (PIDs) to total parts is more than .90% and is called "simple" if the ratio is below .25%. Write a query which returns one row with three columns: the first column should be the total revenue generated by complex items, the second should be the total revenue generated by simple items and the third should be the revenue generated by items which are neither simple nor complex.

```
select
        sum( case when rat > .9 then amt else 0 end ) as TR_complext
        , sum( case when rat < .25 then amt else 0 end) as TR_Simple
        , sum( case when rat <=.9 and rat >= .25 then amt else 0 end) as TR_neither
from
        (select IID, count(1)::float / sum( NoPart) as rat from Assembly
        group by 1) as lhs
left join
        Sales
using(IID)
```

18. Return the long version of the query above. This time there will be two columns and three rows. An item is called "complex" if the ratio of unique parts (PIDs) to total parts is more than .90% and is called "simple" if the ratio is below .25%. The first column should be equal to "Complex", "Simple" or "Neither" and the the second column should be the revenue generated by that type.

```
select
        case
                when rat > .9 then 'Complex'
                when rat < .25 then 'Simple'
                else 'Neither'
        end as typ
        , sum( amt )
from
        (select IID, count(1)::float / sum( NoPart) as rat from Assembly) as lhs
left join
        Sales
using(IID)
group by 1;
```

19. Which salesperson (SalesID) generated the highest profit (total revenue - total cost)?

```
select
        SalesID
from
        (select sum(NoPart * Price ) as totalCost, IID
               from Assembly left join Part using(PID) group by 2) as lhs
left join
        (select sum( rev ) as SR, IID, SalesID, count(1) as ct from sales group by 2,3) as lhs
using( IID )
group by 2
order by (SR - ct * totalCost) desc
limit 1;
```

20. The company believes that there is an assembly cost of roughly $1.00 per part. Note that if an item requires four of the same part, this means that the assembly cost is $4.00. Factoring in this cost, what item has the highest total profit?

```
select
        IID
from
        (select sum(NoPart * Price) + sum( NoPart)*1.0 as totalCost, IID
               from Assembly left join Part using(PID) group by 2) as lhs
left join
        (select sum( rev ) as SR, IID, count(1) as ct from sales group by 2) as lhs
using( IID )
group by 2
order by (SR - ct * totalCost) desc
limit 1;
```

21. For each item (IID), return the average number of days between sales (note that subtracting two dates yields the number of days between those dates).

```
select
        IID, avg(diff) as avgdiff
from
        (select
                IID
                ,dt -  lag(dt) over(partition by IID order by dt asc) as diff
        from
                sales ) as IQ
group by 1;
```

22. Using a cross join return a dataset which contains 3 columns: IID, month and the number of days that that IID was *not* sold that month. Assume that (a) the Sales table contains all information from 2011 and (b) that every possible sales date is represented in the Sales table. Note that if an item is not sold at all during the year it *should* still be in this result.

```
select
        IID, date_part('month', dt ) as mnth
        , sum( case when sales.dt is null then 1 else 0 end ) as daysmissing
from
        (select distinct IID from item) as lhs
cross join
        (select distinct dt from sales) as rhs1
left join
        sales
on lhs.IID = sales.IID and rhs1.dt = sales.dt
group by 1,2;
```

23. Write a query which returns 3 columns: date, the total revenue on that date and the average revenue from the last three days, but *not* including the current day. In other words if today is 1/5, then it should be the average revenue from 1/2, 1/3 and 1/4.

```
select
        dt, sr
        , avg( sr ) over (order by dt asc rows between 4 preceding and 1 preceding) as MA
from
        (select
                sum( rev) as SR, dt
        from
                sales
        group by 2 ) as iq
```

24. Which salesperson (name) sold the most number of *parts*. Note that this is not asking for number of *items* that each salesperson sold, but the number of parts contained within those items.

```
select
        Name
from
        SP
left join
        Sales
        using( salesid)
left join
        Assembly
        using( IID )
group by 1
order by sum(NoPart) desc
limit 1;
```

25. **Bonus** The salespeople in the table are characters from a movie. Which one?