Give SQL queries for the following algebra expressions

\[ \pi_{\text{Ename}}(\sigma_{\text{Eid}=5}(\text{Employee})) \]

\[
\begin{align*}
\text{select} & \quad \text{Ename} \\
\text{from} & \quad \text{Employee} \\
\text{where} & \quad \text{Eid}=5
\end{align*}
\]

\[ \pi_{\text{Eid}}(\text{Employee}) - \pi_{\text{Eid}}(\text{Works}) \]

\[
\begin{align*}
\text{select} & \quad \text{Eid from Employee} \\
& \quad \text{except} \\
\text{select} & \quad \text{Eid from Works}
\end{align*}
\]

\[ \pi_{\text{Ename}}[\rho(\text{E, Employee}) \Join \text{Eid} (\pi_{\text{Eid}}(\text{Employee}) - \pi_{\text{Eid}}(\text{Works}))] \]

\[
\begin{align*}
\text{select} & \quad \text{Ename from Employee} \\
\text{where not exists} & \quad (\text{select} \quad *) \\
& \quad \text{from} \quad \text{Works} \\
& \quad \text{where} \quad \text{Employee.Eid=Works.Eid}
\end{align*}
\]
Exercise #1 – Part B

Employee (Eid, Ename, Salary),
Department (Did, Dname, Eid),
Works (Did, Eid)

\[ \pi_{\text{Employee.Ename, Department.Eid}} (\text{Employee JOIN}_{Eid} \text{ Works JOIN}_{Did} \text{ Department}) \]

\[ \text{select Employee.Ename, Department.Eid} \]
\[ \text{from Employee, Works, Department} \]
\[ \text{where Employee.Eid=Works.Eid and Works.Did=Department.Did} \]

\[ \rho(E1, \text{Employee}), \rho(E2, \text{Employee}) \]
\[ \pi_{\text{Employee.Ename}} (\text{Employee JOIN}_{Eid} (\pi_{E1.Eid} \text{ E1} - (\pi_{E1.Eid} (\sigma_{E1.Salary<E2.Salary} \text{ E1 x E2})))) \]

\[ \text{select E1.name} \]
\[ \text{from Employee as E1} \]
\[ \text{where E1.salary} \geq \text{ all ( select E2.salary} \]
\[ \text{from Employee as E2) }\]
• Motivation: Group by permits us to display aggregate results (e.g., max, min, sum) for groups
  - For instance, if we have group by X, we will get a result for every different value of X
• Recall that aggregate queries without group by return just a single number

• If we put an attribute in select, the attribute must also appear in group by. The opposite is not true. there may be attributes in group by that do not appear in select.

• Any condition that appears in where, is applied before the formation of groups – in other words, records that do not pass the where condition are eliminated before the formation of groups.

• Any condition that appears in having refers to the groups and is applied after the formation of the groups. The condition must involve aggregate functions, or attributes that appear in the select or group by lines.
Exercise #2 – Q1

Q1: Find the total number of copies in stock for each poet

<table>
<thead>
<tr>
<th>poet</th>
<th>book</th>
<th>copies_in_stock</th>
</tr>
</thead>
<tbody>
<tr>
<td>Douglas Livingstone</td>
<td>The Skull</td>
<td>21</td>
</tr>
<tr>
<td>Douglas Livingstone</td>
<td>A Littoral Zone</td>
<td>2</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>Tsetelo</td>
<td>3</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>Must Weep</td>
<td>8</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>A Tough Tale</td>
<td>2</td>
</tr>
</tbody>
</table>

```
select poet, sum(copies_in_stock) as sum
from Writer
group by poet
```
Exercise #2 - Q2

Q2: For each poet, find the max, min, avg and total number of copies in stock

<table>
<thead>
<tr>
<th>poet</th>
<th>book</th>
<th>copies_in_stock</th>
</tr>
</thead>
<tbody>
<tr>
<td>Douglas Livingstone</td>
<td>The Skull</td>
<td>21</td>
</tr>
<tr>
<td>Douglas Livingstone</td>
<td>A Littoral Zone</td>
<td>2</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>Tstetlo</td>
<td>3</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>Must Weep</td>
<td>8</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>A Tough Tale</td>
<td>2</td>
</tr>
</tbody>
</table>

```
select poet, max(copies_in_stock) as max, min(copies_in_stock) as min,
    avg(copies_in_stock) as avg, sum(copies_in_stock) as sum
from Writer
group by poet
```
Exercise #2 – Q3

Q3: For each poet, find the max, min, avg and total number of copies in stock – take into account only books that have > 5 copies in stock

<table>
<thead>
<tr>
<th>poet</th>
<th>book</th>
<th>copies_in_stock</th>
</tr>
</thead>
<tbody>
<tr>
<td>Douglas Livingstone</td>
<td>The Skull</td>
<td>21</td>
</tr>
<tr>
<td>Douglas Livingstone</td>
<td>A Littoral Zone</td>
<td>2</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>Tsetlo</td>
<td>3</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>Must Weep</td>
<td>8</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>A Tough Tale</td>
<td>2</td>
</tr>
</tbody>
</table>

```
select poet, max(copies_in_stock) as max, min(copies_in_stock) as min, avg(copies_in_stock) as avg, sum(copies_in_stock) as sum
from Writer
where copies_in_stock > 5
group by poet
```

<table>
<thead>
<tr>
<th>poet</th>
<th>max</th>
<th>min</th>
<th>avg</th>
<th>sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Douglas Livingstone</td>
<td>21</td>
<td>21</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
</tbody>
</table>
Q4: Find the total number of copies in stock for each poet who has a total of more than 20 copies in stock

<table>
<thead>
<tr>
<th>poet</th>
<th>book</th>
<th>copies_in_stock</th>
</tr>
</thead>
<tbody>
<tr>
<td>Douglas Livingstone</td>
<td>The Skull</td>
<td>21</td>
</tr>
<tr>
<td>Douglas Livingstone</td>
<td>A Littoral Zone</td>
<td>2</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>Tsetlo</td>
<td>3</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>Must Weep</td>
<td>8</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>A Tough Tale</td>
<td>2</td>
</tr>
</tbody>
</table>

**SQL Query:**

```sql
select poet, sum(copies_in_stock) as sum
from Writer
group by poet
having sum>20
```

<table>
<thead>
<tr>
<th>poet</th>
<th>sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Douglas Livingstone</td>
<td>23</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>13</td>
</tr>
</tbody>
</table>
Exercise #2 – Q5

Q5: Find the total number of copies in stock for each poet who has a total of more than 20 copies in stock – take into account only books that have more than 5 copies in stock.

<table>
<thead>
<tr>
<th>poet</th>
<th>book</th>
<th>copies_in_stock</th>
</tr>
</thead>
<tbody>
<tr>
<td>Douglas Livingstone</td>
<td>The Skull</td>
<td>21</td>
</tr>
<tr>
<td>Douglas Livingstone</td>
<td>A Littoral Zone</td>
<td>2</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>Tstetlo</td>
<td>3</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>Must Weep</td>
<td>8</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>A Tough Tale</td>
<td>2</td>
</tr>
</tbody>
</table>

SQL Code:
```
SELECT poet, SUM(copies_in_stock) AS sum
FROM Writer
WHERE copies_in_stock > 5
GROUP BY poet
HAVING sum > 20
```

<table>
<thead>
<tr>
<th>poet</th>
<th>sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Douglas Livingstone</td>
<td>21</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>8</td>
</tr>
</tbody>
</table>
Q6: Find the total number of copies in stock for each poet whose name starts with any letter after “E”

<table>
<thead>
<tr>
<th>poet</th>
<th>book</th>
<th>copies_in_stock</th>
</tr>
</thead>
<tbody>
<tr>
<td>Douglas Livingstone</td>
<td>The Skull</td>
<td>21</td>
</tr>
<tr>
<td>Douglas Livingstone</td>
<td>A Littoral Zone</td>
<td>2</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>Tstetlo</td>
<td>3</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>Must Weep</td>
<td>8</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>A Tough Tale</td>
<td>2</td>
</tr>
</tbody>
</table>

Solution #1

```sql
select poet, sum(copies_in_stock) as sum
from Writer
where poet > "E"
group by poet
```

Solution #2

```sql
select poet, sum(copies_in_stock) as sum
from Writer
group by poet
having poet > "E"
```
Q7: Find the total number of copies in stock for each poet who has more than 2 books

<table>
<thead>
<tr>
<th>poet</th>
<th>book</th>
<th>copies_in_stock</th>
</tr>
</thead>
<tbody>
<tr>
<td>Douglas Livingstone</td>
<td>The Skull</td>
<td>21</td>
</tr>
<tr>
<td>Douglas Livingstone</td>
<td>A Littoral Zone</td>
<td>2</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>Tstetlo</td>
<td>3</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>Must Weep</td>
<td>8</td>
</tr>
<tr>
<td>Mongane Wally</td>
<td>A Tough Tale</td>
<td>2</td>
</tr>
</tbody>
</table>

select poet, sum(copies_in_stock) as sum
from Writer
group by poet
having count(*)>2

What will be the result of this query?
select poet
from Writer
group by poet
having count(*)>2

<table>
<thead>
<tr>
<th>poet</th>
<th>sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mongane Wally</td>
<td>13</td>
</tr>
</tbody>
</table>
Exercise #3 – Part A

Sailors (sid, sname, rating, age),
Reserves (sid, bid, date),
Boats (bid, bname, color)

Find the number of sailor records

```
select count(*)
from Sailors
```

Find the average age of sailors with rating 10

```
select avg(S.age)
from Sailors as S
where S.rating=10
```

Find how many different ratings exist for sailors who are called Bob

```
select count(distinct S.rating)
from Sailors as S
where S.sname='Bob'
```
Exercise #3 – Part B

Find name and age of the oldest sailor(s)

**Correct solution**

```sql
select S.sname, S.age
from Sailors as S
where S.age =
    (select max(S2.age)
     from Sailors as S2)
```

Illegal! If there is no group by, we can only have aggregates (without attributes).
Exercise #3 – Part C

For each red boat, display the bid and the number of reservations for this boat

```
select  B.bid, count(*) as scount
from    Boats as B, Reserves as R
where   R.bid=B.bid and B.color='red'
group by B.bid
```

For each red boat, display the name and the number of reservations for this boat

```
select  B.bname, count(*) as scount
from    Boats as B, Reserves as R
where   R.bid=B.bid and B.color='red'
group by B.bid, B.bname
```
Find the age of the youngest sailor with age > 18, for each rating with at least 2 such sailors.
Exercise #3 – Part E

Find the age of the youngest sailor with age > 18, for each rating with at least 2 sailors (of any age)

```sql
select S.rating, min(S.age)
from Sailors as S
where S.age > 18
group by S.rating
having 1 < (select count(*)
            from Sailors as S2
            where S.rating = S2.rating)
```
Exercise #3 – Part F

Find those ratings for which the average age is the minimum over all ratings

```
select Temp.rating, Temp.avgage
from (select S.rating, avg(S.age) as avgage
     from Sailors as S
     group by S.rating) as Temp
where Temp.avgage = (select min(Temp.avgage) from Temp)
```