# Database Programming

Issues in Aggregates

# Introduction

- Aggregate functions are a very important element of SQL programming
- Using these functions on the simple level does not have any problems

```
SELECT count(*) from Student
```

- However, when the problems become more complex, there are certain problems, which have to be avoided in order to show a proper result

# Headers for Aggregate Fields

- Select all international students grouping them by first letter of their last name

# Stage One

- Select all students grouped by last name

```
SELECT Left(lastName,1), count(Left(lastName,1))
From Student s inner join person p
on s.number = p.number
GROUP BY (Left(lastName, 1))
```

# Results

```
---- -----------
A    17
B    34
C    32
D    24
E    8
F    18
G    17
H    22
I    4
J    10
K    23
L    25
M    36
N    6
O    7
P    26
R    18
S    33
T    20
U    1
V    11
W    19
Y    1
Z    8

(24 row(s) affected)
```

Problems to overcome:

- No column headers
- What will happen when only international students are selected?

# Attempt to fix the Problems

```
SELECT Left(lastName,1) Letter,
    count(Left(lastName,1)) [Count of Letter]
From Student s inner join person p
on s.number = p.number
WHERE s.IsInternational = 1
GROUP BY (Left(lastName, 1))
```

- New Problem: Lost 7 rows, which were present

Results

| Letter | Count of Letter |
|---|---|
| A | 3 |
| B | 2 |
| C | 2 |
| D | 3 |
| F | 2 |
| G | 2 |
| H | 2 |
| J | 1 |
| K | 3 |
| L | 5 |
| M | 2 |
| P | 1 |
| R | 4 |
| S | 2 |
| T | 2 |
| W | 3 |
| Z | 5 |

(17 row(s) affected)

# Better Fix

- Use Outer Join
- Place condition in 'ON' clause

```
SELECT Left(p.lastName,1) Letter,
    count(s.number)  [Count of Letter]
From  Student s RIGHT OUTER JOIN Person p
on s.number = p.number and s.isInternational=1
GROUP BY (Left(p.lastName, 1))
```

- Result

```
Letter Count of Lette
------ --------------
A      3
B      2
C      2
D      3
E      0
F      2
G      2
H      2
I      0
J      1
K      3
L      5
M      2
N      0
O      0
P      1
R      4
S      2
T      2
U      0
V      0
W      3
Y      0
Z      5
Warning: Null value i

(24 row(s) affected)
```

# Problem 2

- Get average amount paid by the students by using Audit table
- Additional problem: some students paid more than once therefore sum of all payments needs to be calculated

# First Stage

- Get Sum of payments for all students

```
SELECT p.lastName, p.firstName,
            sum(a.amount) as total
FROM Person p Inner JOIN Audit a
ON p.number = a.studentNumber
Group By  p.lastName, p.firstName
```

- Result

```
total
--------------------
1000.00
5291.44
5291.44
1000.00
6037.32
6791.44
1159.00
3000.00
1000.00
10000.00
13159.00
5291.44
2000.00
13159.00
1000.00
```

# Problems with applying Avg

- We cannot apply one aggregate function to another aggregate function

```
Msg 130, Level 15, State 1, Line 2
Cannot perform an aggregate function on an expression containing an aggregate or a subquery.
```

- We cannot apply aggregate to a subquery (for the same reason)
- Solution: Create a local table and apply average to it

# Proper Solution

- Create a table ‘MySumTable’ and select average of all totals from it

```
SELECT AVG(total) FROM (SELECT p.lastName, p.firstName,
            (sum(a.amount)) as total
FROM Person p Inner JOIN Audit a
ON p.number = a.studentNumber
Group By  p.lastName, p.firstName) MySumTable
```

# Problem 3

- I would like to know all students who received the marks, which exceed the minimum mark for less than 30% of it for Fall 2010.
- I need to get first and last name of the students, name of the course for which it happened and (obviously) their mark

# Solution

```
select p.lastname, p.firstname, co.courseNumber, ct.finalMark
from person p inner join coursestudent ct on p.number = ct.studentNumber
inner join courseoffering co on co.id = ct.CourseOfferingId
WHERE co.sessionCode = 'F10'
GROUP BY p.lastName, p.firstName, ct.FinalMark, co.courseNumber, co.sessionCode
HAVING ct.finalMark < (SELECT MIN(finalMark)*1.3 FROM CourseStudent cs1
                            INNER JOIN CourseOffering co1 on cs1.CourseOfferingId = co1.id
                       WHERE co1.courseNumber = co.courseNumber and co1.SessionCode=co.sessionCode)
Order by co.courseNumber
```

# Comments

- Results of aggregate functions can be used in any equation
- Try to get the numbers right first because the results of the application of aggregates are not obvious
- I start with the subquery and then apply it

```
SELECT MIN(finalMark) FROM CourseStudent cs1
                        INNER JOIN CourseOffering co1 on cs1.CourseOfferingId = co1.id
                    WHERE co1.courseNumber = 'PROG8090' and co1.SessionCode='F10'
```

# Bonus Problem

- Try to use Payment table to figure out which student paid less than average in tuition

# Solution

```
SELECT s.number, p.lastName, p.firstName, sum(py.amount)
FROM Student s INNER JOIN Person p
ON p.number = s.number
INNER JOIN Payment py ON py.studentNumber = s.number
GROUP BY s.number, p.lastName, p.firstName
HAVING SUM(py.amount) < (SELECT avg(total) FROM
                            (SELECT SUM(py.amount) as total FROM Payment py INNER JOIN Student s
                                ON py.studentNumber = s.number
                                GROUP by s.number) as MyTable)
```