IS 441 July 31 Class Recapture: Basic SELECT Statement

|  |  |
| --- | --- |
| SELECT restaurantID, annualsalesFROM RestaurantsWHERE annualSales >= 350000;“350 K or more” | If I want more columns: Type of service, Date openedSELECT restaurantID, annualsales, TypeOfService, DateOpenedFROM RestaurantsWHERE annualSales >= 350000; |
| What if I mean: “more than 350K”?WHERE annualSales > 350000; | All columns: SELECT \*(Shorter result: annualSales >= 600000 )SELECT \*FROM RestaurantsWHERE annualSales >= 600000; |

**Column alias:**

Good for display ONLY (not in calculation; not in condition – WHERE; not in ORDERing)

SELECT RestaurantID AS RID, AnnualSales AS Sales

FROM Restaurants

WHERE AnnualSales >=500000

**Expression (mathematical formulas)**

1. Calculations:

- +-\*/ and other math operations

Example 1: If I want to display the sales plus 10%:

SELECT RestaurantID AS RID, AnnualSales, **AnnualSales \* 1.1** AS SalesPlus10Percent

FROM Restaurants

WHERE AnnualSales >=500000

Example 2: If I want to display the sales per foot of a restaurant

Key expression (formula) is:

AnnualSales/Squarefootage AS SalesPerFt

SELECT restaurantID, annualsales, Squarefootage

AnnualSales/Squarefootage AS SalesPerFt

FROM Restaurants

WHERE annualSales >= 600000;

Summary: In SELECT-clause, in addition to the existing fields, we can also add expressions (forumals) to be displayed.

1. Using functions (AVG, SUM, etc)

|  |  |
| --- | --- |
| SELECT AVG(annualsales) AS AvgSales, AVG(Squarefootage) AS AvgSqFtFROM RestaurantsComment: These AVG s were calculated/conducted over the scope of ALL restaurants.(Output below, for ease of comparison of codes – ONLY in this example; NOT the HW format) | SELECT AVG(annualsales) AS AvgSales, AVG(Squarefootage) AS AvgSqFtFROM RestaurantsWHERE AnnualSales >= 600000Comment: These AVG s were calculated/conducted over the scope of Restaurants WHOSE SALES >=600000 |
|  |  |

Showcase the error of mixing row values and set values (or: fields and aggregate functions)



In the same logic, the corresponding Restaurants examples would be:”

SELECT AnnualSales – AVG(AnnualSales)

FROM Restaurants;

\*\*\* Mixture of row and set values: NO-NO

 Example: ONLY row values, no problem

|  |  |
| --- | --- |
| SELECT RestaurantID, AnnualSales, SquarefootageFROM Restaurants; |  |

Example: ONLY set values (aggregate functions), no problem:

|  |  |
| --- | --- |
| SELECT COUNT(RestaurantID) AS NumRest, AVG(AnnualSales) AS AVGSales, AVG(Squarefootage) AS AVGSqFtFROM Restaurants; |  |

Example: Mixture – error msg:

|  |  |
| --- | --- |
| SELECT RestaurantID, AVG(AnnualSales) AS AVGSales, AVG(Squarefootage) AS AVGSqFtFROM Restaurants; |  |

ORDER BY

Example: I want to list restaurants, ordered by their annual sales

SELECT RestaurantID, AnnualSales, Squarefootage FROM Restaurants ORDER BY AnnualSales;

|  |  |
| --- | --- |
|  | Comments:1. Sorted by AnnualSales, low to high
2. Note: R02 and R08, w sales at 350K, SqFt low to high (1000, then 2850)
3. Note: R03, R19, and R10 are all at 750K, with SqFt hi-to-low (2500 – 2000 – 1000)
 |

Annual sales high to low: use DESC (= descending)

SELECT RestaurantID, AnnualSales, Squarefootage FROM Restaurants ORDER BY AnnualSales DESC;

New task: For those with same sales, I want them further sourted on SqFt, low to high.

SELECT RestaurantID, AnnualSales, Squarefootage FROM Restaurants

Sorting on two fields: separated by “,”

ORDER BY AnnualSales DESC, Squarefootage;

If there is no “DESC” in ORDER BY, default is low to high (ascending)



GROUP BY

Task: I want to see the AVG sales, AVG SqFt, and total number of restaurants in EACH CITY.

Solution: GROUP BY City – gives the aggregate function values for each city AS A WHOLE (there’s no more individual restaurants: their values are COLLAPSED in their city group).

The Agg Func: AVG(AnnualSales), AVG(SquareFootage), COUNT(RestaurantID)

SELECT City, AVG(AnnualSales) AS vAvgSale, AVG(SquareFootage) As AvgSqFt, COUNT(RestaurantID) AS NumRest FROM Restaurants GROUP BY City;



SELECT AVG(AnnualSales) AS vAvgSale, AVG(SquareFootage) As AvgSqFt, COUNT(RestaurantID) AS NumRest FROM Restaurants;

What’s the outcome? One value for each agg func, for all rest in the table.



Important Point:

ORDER BY vs GROUP BY

|  |  |  |  |
| --- | --- | --- | --- |
|  | ORDER BY | GROUP BY | Remarks |
| Logical Action | Sort: Indivi rows are still there | Collapse: Indiv rows disappear; only groups |  |
| “Mandatory” or not | Don’t have to select the field to ORDER | Don’t have to select the field to GROUP | But you don’t see whether you did it right or not |
| In the case of mixture of row & set values | N/A | If a field is the GROUP BY field, that field can appear in SELECT | If a field is the GROUP BY field, does it have to be in SELECT? - NO |
| Multi-field | Output 1st sorted by the 1st ORDER BY field, then by the 2nd ORDER BY field;The multi-fields are separated with “,”The M-field sort can be different – one ASC, another DESC, etc | “Almost N/A” – logically can do but messier | In ORDER BY, the default is ASC – don’t need to type “ASC”; if descending is desired, use DESC |
|  |  |  |  |

HAVING as conditions for groups:

I want to see the cities with more than 2 restaurants in the city, with their AVG…

SELECT City, AVG(AnnualSales) AS vAvgSale, AVG(SquareFootage) As AvgSqFt, COUNT(RestaurantID) AS NumRest FROM Restaurants GROUP BY City

HAVING COUNT(RestaurantID) >2;

