Database Design and Implementation

HW 06

Team 08

Henriod, Terence

Santoyo, Jorge

Singh, Raja

March 11, 2015

/\*\*

\* IS675 - HW06 - Team08

\* Using SQL Operations with More Complex Queries

\*

\* Division of Labor

\* Well...

\*/

/\*\*

\* Exercise 01

\*

\* Summarize the actual hours worked and labor cost (hours worked \* hourly

\* pay rate) by JobID and TaskID for all rows in the JobTask table. Remember

\* that an employee's pay rate changes by date, so it is necessary to locate

\* the correct pay rate by date as you did for question #15 for HW#5. If you

\* were not able to get question #15 on HW#5 to work, I recommend doing that

\* before starting on this question. Compare the actual labor hours to the

\* estimated labor hours, and the actual labor costs to the estimated

\* labor costs.

\*/

DROP VIEW [v\_TimeSheetsWithPay];

CREATE VIEW [v\_TimeSheetsWithPay] AS

SELECT

[TimeSheet].[EmpID] AS 'EmpID',

[Employee].[LastName] + ', ' + [Employee].[FirstName] AS 'Employee Name',

[TimeSheet].[JobID] AS 'JobID',

[TimeSheet].[TaskID] AS 'TaskID',

[Job].[JobCompleted] AS 'JobCompleted',

[TimeSheet].[StartWork] AS 'StartWork',

[TimeSheet].[HoursWorked] AS 'HoursWorked',

[EmployeePay].[HourlyPayRate] AS 'HourlyPay',

[TimeSheet].[HoursWorked] \* [EmployeePay].[HourlyPayRate] AS 'LaborCost'

FROM

[TimeSheet]

INNER JOIN [EmployeePay] ON

[TimeSheet].[EmpID] = [EmployeePay].[EmpID]

INNER JOIN [Employee] ON

[TimeSheet].[EmpID] = [Employee].[EmpID]

INNER JOIN [Job] ON

[TimeSheet].[JobID] = [Job].[JobID]

WHERE

[EmployeePay].[DateStartPay] <= [TimeSheet].[StartWork] AND

[TimeSheet].[StartWork] < ISNULL([EmployeePay].[DateEnd], GETDATE()) AND

[TimeSheet].[JobID] IS NOT NULL

;

DROP VIEW [v\_ActualLaborByJobTask];

CREATE VIEW [v\_ActualLaborByJobTask] AS

SELECT

[v\_TimeSheetsWithPay].[JobID] AS 'JobID',

[v\_TimeSheetsWithPay].[TaskID] AS 'TaskID',

SUM([v\_TimeSheetsWithPay].[HoursWorked]) AS 'HoursWorked',

SUM(

[v\_TimeSheetsWithPay].[HoursWorked] \* [v\_TimeSheetsWithPay].[HourlyPay]

) AS 'LaborCost'

FROM

[v\_TimeSheetsWithPay]

GROUP BY

[v\_TimeSheetsWithPay].[JobID],

[v\_TimeSheetsWithPay].[TaskID]

;

DROP VIEW [v\_LaborCostComparisons];

CREATE VIEW [v\_LaborCostComparisons] AS

SELECT

[JobTask].[JobID] AS 'JobID',

[JobTask].[TaskID] AS 'TaskID',

[Task].[TaskDescription] AS 'TaskDescription',

CONVERT(varchar, [JobTask].[DateStarted], 101) AS 'DateStarted',

ISNULL(

CONVERT(varchar, [JobTask].[DateCompleted], 101),

'Not Done'

) AS 'DateCompleted',

[JobTask].[EstHours] AS 'EstHours',

ISNULL([v\_ActualLaborByJobTask].[HoursWorked], 0.00) AS 'ActualHoursWorked',

[JobTask].[EstHours] - ISNULL([v\_ActualLaborByJobTask].[HoursWorked], 0.00)

AS 'LaborHoursVariance',

[JobTask].[EstLaborCost] AS 'EstLaborCost',

ISNULL([v\_ActualLaborByJobTask].[LaborCost], 0.00) AS 'ActualLaborCost',

[JobTask].[EstLaborCost] -

ISNULL([v\_ActualLaborByJobTask].[LaborCost], 0.00) AS 'LaborCostVariance'

FROM

[JobTask]

INNER JOIN [Task] ON

[JobTask].[TaskID] = [Task].[TaskID]

LEFT OUTER JOIN [v\_ActualLaborByJobTask] ON

[JobTask].[JobID] = [v\_ActualLaborByJobTask].[JobID] AND

[JobTask].[TaskID] = [v\_ActualLaborByJobTask].[TaskID]

;

SELECT

\*

FROM

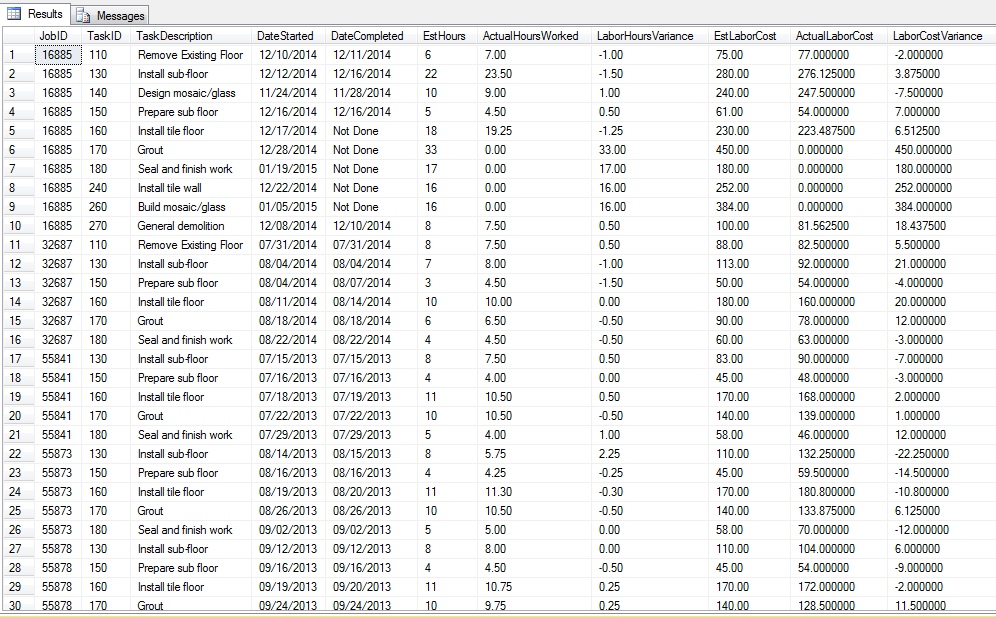
[v\_LaborCostComparisons]

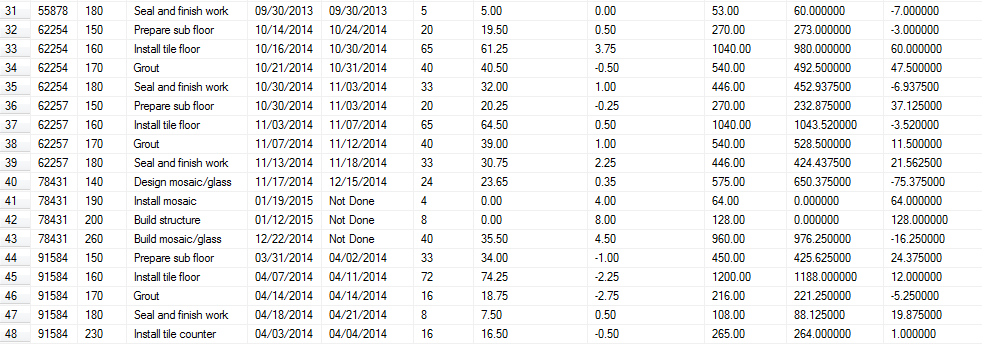
ORDER BY

[v\_LaborCostComparisons].[JobID],

[v\_LaborCostComparisons].[TaskID]

;





/\*\*

\* Exercise 02

\*

\* Summarize the actual material costs by jobID and task ID and compare

\* them to the estimated material cost for each row in the JobTask table.

\* This is very similar to what you did for question #11 in HW#5, so this

\* should be fairly easy if you got question #11 to work.

\*/

DROP VIEW [v\_ActualMaterialsByTask];

CREATE VIEW [v\_ActualMaterialsByTask] AS

SELECT

[MaterialAssigned].[JobID] AS 'JobID',

[MaterialAssigned].[TaskID] AS 'TaskID',

SUM([MaterialAssigned].[Quantity] \* [MaterialPurchased].[CostPerUOM]) AS 'MaterialCost'

FROM

[MaterialAssigned]

INNER JOIN [MaterialPurchased] ON

[MaterialAssigned].[POID] = [MaterialPurchased].[POID]

GROUP BY

[MaterialAssigned].[JobID],

[MaterialAssigned].[TaskID]

;

DROP VIEW [v\_MaterialCostComparisons];

CREATE VIEW [v\_MaterialCostComparisons] AS

SELECT

[JobTask].[JobID] AS 'JobID',

[JobTask].[TaskID] AS 'TaskID',

[Task].[TaskDescription] AS 'TaskDescription',

CONVERT(varchar, [JobTask].[DateStarted], 101) AS 'DateStarted',

ISNULL(

CONVERT(varchar, [JobTask].[DateCompleted], 101),

'Not Done'

) AS 'DateCompleted',

[JobTask].[EstMaterialCost] AS 'EstMaterialCost',

ISNULL([v\_ActualMaterialsByTask].[MaterialCost], 0.00) AS 'ActualMaterialCost',

(

[JobTask].[EstMaterialCost] -

ISNULL([v\_ActualMaterialsByTask].[MaterialCost], 0.00)

) AS 'MaterialCostVariance'

FROM

[JobTask]

INNER JOIN [Task] ON

[JobTask].[TaskID] = [Task].[TaskID]

LEFT OUTER JOIN [v\_ActualMaterialsByTask] ON

[JobTask].[JobID] = [v\_ActualMaterialsByTask].[JobID] AND

[JobTask].[TaskID] = [v\_ActualMaterialsByTask].[TaskID]

;

SELECT

\*

FROM

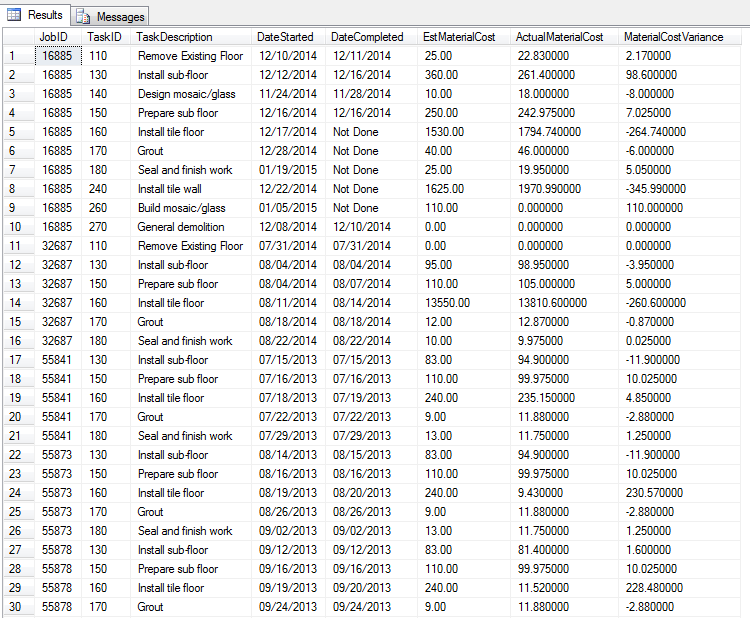
[v\_MaterialCostComparisons]

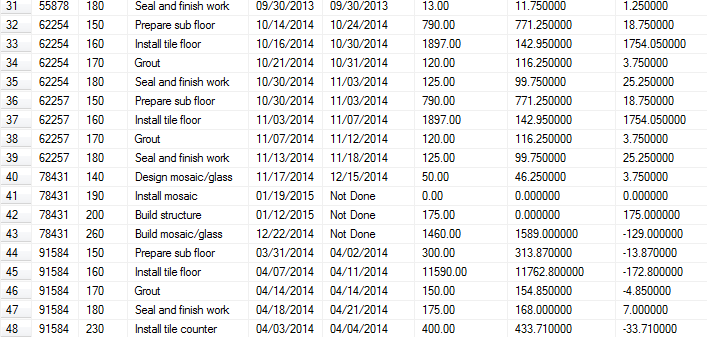
ORDER BY

[v\_MaterialCostComparisons].[JobID],

[v\_MaterialCostComparisons].[TaskID]

;





/\*\*

\* Exercise 03

\*

\* Now it’s time to put them together. Compare actual to estimated costs

\* for each row in the JobTask table. The PercentVariance is the percentage

\* variance between the TotalEstCost and the TotalActualCost. The

\* general calculation is:

\* ((TotalEstCost – TotalActualCost)/TotalEstCost) \* 100.

\*/

CREATE VIEW [v\_TotalCostsByTask] AS

SELECT

[v\_LaborCostComparisons].[JobID] AS 'JobID',

[v\_LaborCostComparisons].[TaskID] AS 'TaskID',

[v\_LaborCostComparisons].[TaskDescription] AS 'TaskDescription',

[v\_LaborCostComparisons].[DateStarted] AS 'DateStarted',

[v\_LaborCostComparisons].[DateCompleted] AS 'DateCompleted',

(

[v\_LaborCostComparisons].[EstLaborCost] +

[v\_MaterialCostComparisons].[EstMaterialCost]

) AS 'TotalEstCost',

(

[v\_LaborCostComparisons].[ActualLaborCost] +

[v\_MaterialCostComparisons].[ActualMaterialCost]

) AS 'TotalActualCost'

FROM

[v\_LaborCostComparisons]

INNER JOIN [v\_MaterialCostComparisons] ON

[v\_LaborCostComparisons].[JobID] = [v\_MaterialCostComparisons].[JobID] AND

[v\_LaborCostComparisons].[TaskID] = [v\_MaterialCostComparisons].[TaskID]

;

DROP VIEW [v\_AllCostComparisonsByTask];

CREATE VIEW [v\_AllCostComparisonsByTask] AS

SELECT

[v\_TotalCostsByTask].[JobID] AS 'JobID',

[v\_TotalCostsByTask].[TaskID] AS 'TaskID',

[v\_TotalCostsByTask].[TaskDescription] AS 'Task Description',

[v\_TotalCostsByTask].[DateStarted] AS 'DateStarted',

[v\_TotalCostsByTask].[DateCompleted] AS 'DateCompleted',

[v\_LaborCostComparisons].[EstHours] AS 'EstHours',

[v\_LaborCostComparisons].[ActualHoursWorked] AS 'ActualHoursWorked',

[v\_LaborCostComparisons].[LaborHoursVariance] AS 'LaborHoursVariance',

[v\_LaborCostComparisons].[EstLaborCost] AS 'EstLaborCost',

[v\_LaborCostComparisons].[ActualLaborCost] AS 'ActualLaborCost',

[v\_LaborCostComparisons].[LaborCostVariance] AS 'LaborCostVariance',

[v\_MaterialCostComparisons].[EstMaterialCost] AS 'EstMaterialCost',

[v\_MaterialCostComparisons].[ActualMaterialCost] AS 'ActualMaterialCost',

[v\_MaterialCostComparisons].[MaterialCostVariance] AS 'MaterialCostVariance',

[v\_TotalCostsByTask].[TotalEstCost] AS 'TotalEstCost',

[v\_TotalCostsByTask].[TotalActualCost] AS 'TotalActualCost',

(

[v\_TotalCostsByTask].[TotalEstCost] -

[v\_TotalCostsByTask].[TotalActualCost]

) AS 'TotalCostVariance',

(

(

[v\_TotalCostsByTask].[TotalEstCost] -

[v\_TotalCostsByTask].[TotalActualCost]

) \*

(

100.00 /

[v\_TotalCostsByTask].[TotalEstCost]

)

) AS 'PercentVariance'

FROM

[v\_TotalCostsByTask]

INNER JOIN [v\_LaborCostComparisons] ON

[v\_TotalCostsByTask].[JobID] = [v\_LaborCostComparisons].[JobID] AND

[v\_TotalCostsByTask].[TaskID] = [v\_LaborCostComparisons].[TaskID]

INNER JOIN [v\_MaterialCostComparisons] ON

[v\_TotalCostsByTask].[JobID] = [v\_MaterialCostComparisons].[JobID] AND

[v\_TotalCostsByTask].[TaskID] = [v\_MaterialCostComparisons].[TaskID]

;

SELECT

\*

FROM

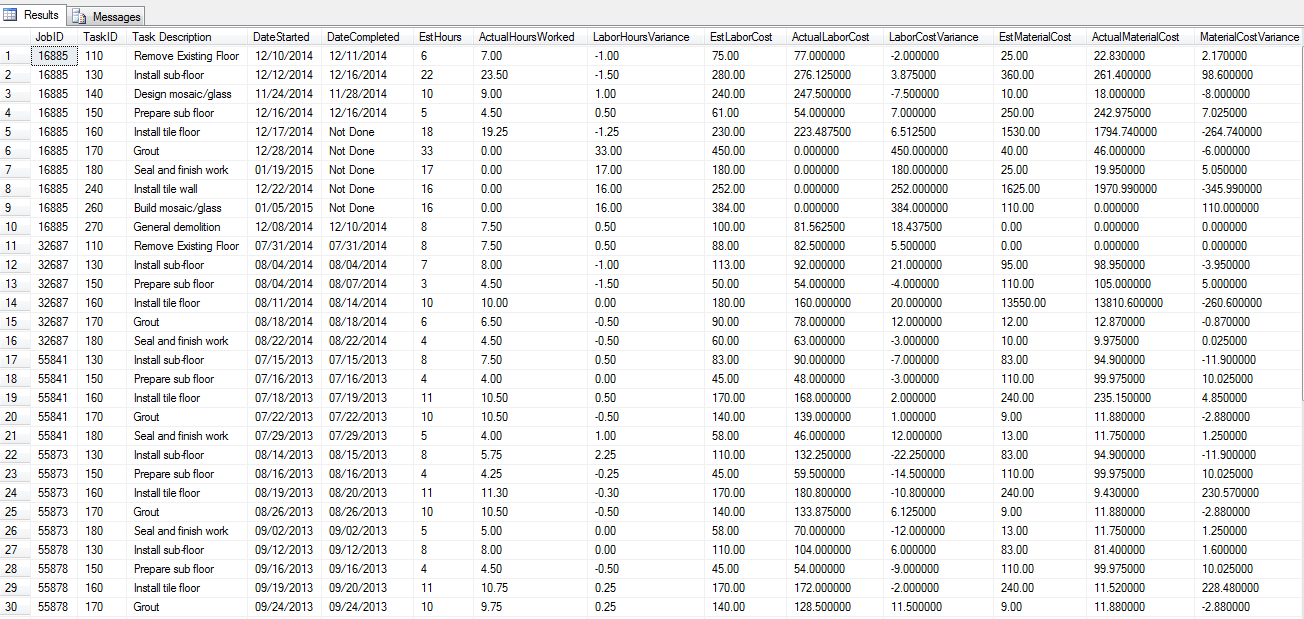
[v\_AllCostComparisonsByTask]

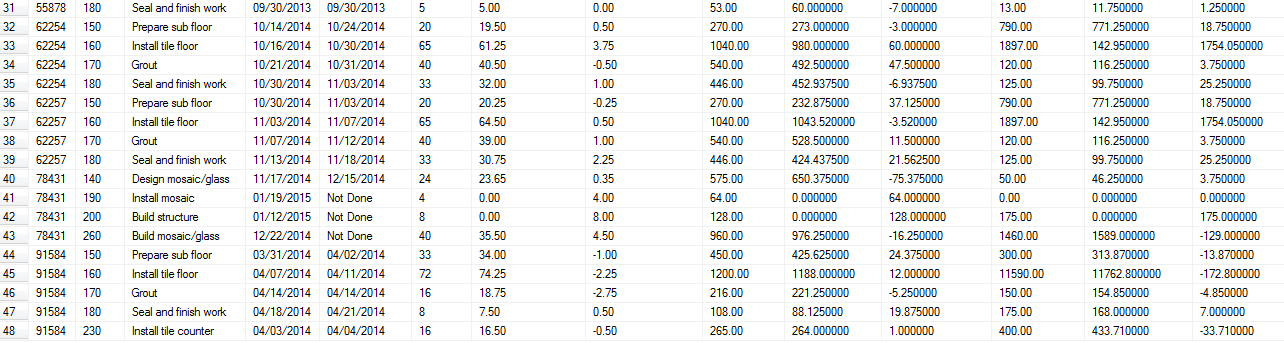
ORDER BY

[v\_AllCostComparisonsByTask].[JobID],

[v\_AllCostComparisonsByTask].[TaskID]

;





/\*\*

\* Exercise 04

\*

\* Summarize the information created in question #3 by job. The result

\* table should have one row per job in the Job table. Add additional data

\* from the Job and Client tables to provide more information about each

\* job in the result table. Hint: \textbf{\underline{Calculate}} the

\* PercentVariance – you cannot sum that field.

\*/

DROP VIEW [v\_TotalCostsByJob];

CREATE VIEW [v\_TotalCostsByJob] AS

SELECT

[v\_AllCostComparisonsByTask].[JobID] AS 'JobID',

SUM([v\_AllCostComparisonsByTask].[EstHours]) AS 'EstHours',

SUM([v\_AllCostComparisonsByTask].[ActualHoursWorked]) AS 'ActualHoursWorked',

SUM([v\_AllCostComparisonsByTask].[LaborHoursVariance]) AS 'LaborHoursVariance',

SUM([v\_AllCostComparisonsByTask].[EstLaborCost]) AS 'EstLaborCost',

SUM([v\_AllCostComparisonsByTask].[ActualLaborCost]) AS 'ActualLaborCost',

SUM([v\_AllCostComparisonsByTask].[LaborCostVariance]) AS 'LaborCostVariance',

SUM([v\_AllCostComparisonsByTask].[EstMaterialCost]) AS 'EstMaterialCost',

SUM([v\_AllCostComparisonsByTask].[ActualMaterialCost]) AS 'ActualMaterialCost',

SUM([v\_AllCostComparisonsByTask].[MaterialCostVariance]) AS 'MaterialCostVariance',

SUM([v\_AllCostComparisonsByTask].[TotalEstCost]) AS 'TotalEstCost',

SUM([v\_AllCostComparisonsByTask].[TotalActualCost]) AS 'TotalActualCost',

SUM([v\_AllCostComparisonsByTask].[TotalCostVariance]) AS 'TotalCostVariance',

(

SUM([v\_AllCostComparisonsByTask].[TotalCostVariance]) \* 100 /

SUM([v\_AllCostComparisonsByTask].[TotalEstCost])

) AS 'PercentVariance'

FROM

[v\_AllCostComparisonsByTask]

GROUP BY

[v\_AllCostComparisonsByTask].[JobID]

;

DROP VIEW [v\_JobInfo];

CREATE VIEW [v\_JobInfo] AS

SELECT

[Job].[JobID] AS 'JobID',

[Job].[JobName] AS 'JobName',

[Client].[ClientName] AS 'ClientName',

CASE

WHEN [Job].[JobCompleted] = 1

THEN 'Finished'

ELSE

'Not Finished'

END /\*\*/ AS 'JobStatus'

FROM

[Job]

INNER JOIN [Client] ON

[Job].[ClientID] = [Client].[ClientID]

;

DROP VIEW [v\_JobSummary];

CREATE VIEW [v\_JobSummary] AS

SELECT

[v\_JobInfo].[JobID] AS 'JobID',

[v\_JobInfo].[JobName] AS 'JobName',

[v\_JobInfo].[ClientName] AS 'ClientName',

[v\_JobInfo].[JobStatus] AS 'JobStatus',

[v\_TotalCostsByJob].[EstHours] AS 'EstHours',

[v\_TotalCostsByJob].[ActualHoursWorked] AS 'ActualHoursWorked',

[v\_TotalCostsByJob].[LaborHoursVariance] AS 'LaborHoursVariance',

[v\_TotalCostsByJob].[EstLaborCost] AS 'EstLaborCost',

[v\_TotalCostsByJob].[ActualLaborCost] AS 'ActualLaborCost',

[v\_TotalCostsByJob].[LaborCostVariance] AS 'LaborCostVariance',

[v\_TotalCostsByJob].[EstMaterialCost] AS 'EstMaterialCost',

[v\_TotalCostsByJob].[ActualMaterialCost] AS 'ActualMaterialCost',

[v\_TotalCostsByJob].[MaterialCostVariance] AS 'MaterialCostVariance',

[v\_TotalCostsByJob].[TotalEstCost] AS 'TotalEstCost',

[v\_TotalCostsByJob].[TotalActualCost] AS 'TotalActualCost',

[v\_TotalCostsByJob].[TotalCostVariance] AS 'TotalCostVariance',

[v\_TotalCostsByJob].[PercentVariance] AS 'PercentVariance'

FROM

[v\_JobInfo]

LEFT OUTER JOIN [v\_TotalCostsByJob] ON

[v\_JobInfo].[JobID] = [v\_TotalCostsByJob].[JobID]

;

SELECT

\*

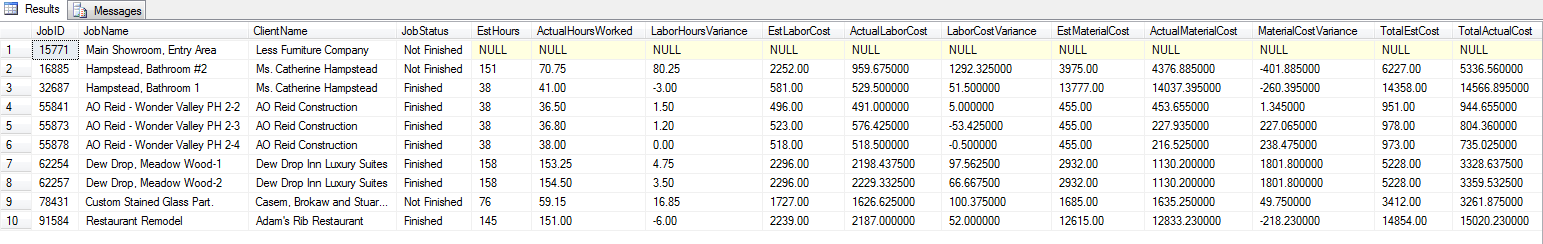
FROM

[v\_JobSummary]

ORDER BY

[v\_JobSummary].[JobID]

;



/\*\*

\* Exercise 05

\*

\* Which job that is \underline{finished} had actual total costs that were

\* closest to the estimated total costs? (PercentVariance closest to zero)

\* Make sure that the query could select the correct job from any data

\* set – the query should not just work with our test data set.

\*/

SELECT

\*

FROM

[v\_JobSummary]

WHERE

[v\_JobSummary].[PercentVariance] = (

SELECT

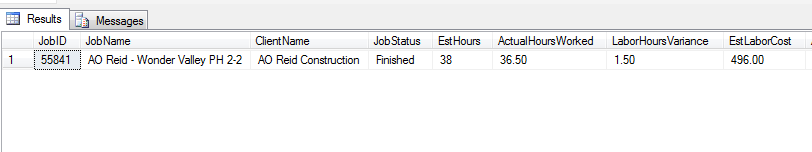
MIN(ABS([v\_JobSummary].[PercentVariance]))

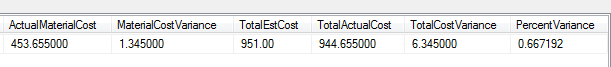
FROM

[v\_JobSummary]

)

;





/\*\*

\* Exercise 06

\*

\* Which job that is \underline{finished} had the largest percentage

\* positive labor hours variance? In other words, which finished job was

\* able to be completed with the least number of labor hours, when compared

\* to the estimated labor hours? The percentage labor hours variance is

\* calculated as the LaborHoursVariance/EstHours \* 100. Add in the name of

\* the employee who served as the manager for the job.

\*/

DROP VIEW [v\_LaborHoursVariance];

CREATE VIEW [v\_LaborHoursVariance] AS

SELECT

[v\_JobSummary].[JobID],

[v\_JobSummary].[JobName],

[v\_JobSummary].[ClientName],

[v\_JobSummary].[EstHours],

[v\_JobSummary].[ActualHoursWorked],

[v\_JobSummary].[EstHours] - [v\_JobSummary].[ActualHoursWorked] AS 'LaborHoursVariance',

([v\_JobSummary].[EstHours] - [v\_JobSummary].[ActualHoursWorked]) \* 100 / [v\_JobSummary].[EstHours] AS 'PercentHoursVariance'

FROM

[v\_JobSummary]

WHERE

[v\_JobSummary].[JobStatus] = 'Finished'

;

DROP VIEW [v\_JobManagerNames];

CREATE VIEW [v\_JobManagerNames] AS

SELECT

[Job].[JobID] AS 'JobID',

[Employee].[LastName] + ', ' + [Employee].[FirstName] AS 'EmployeeManager'

FROM

[Job]

INNER JOIN [Employee] ON

[Job].[EmpManagerID] = [Employee].[EmpID]

;

SELECT

[v\_LaborHoursVariance].[JobID],

[v\_LaborHoursVariance].[JobName],

[v\_LaborHoursVariance].[ClientName],

[v\_JobManagerNames].[EmployeeManager] AS 'EmployeeManager',

[v\_LaborHoursVariance].[EstHours],

[v\_LaborHoursVariance].[ActualHoursWorked],

[v\_LaborHoursVariance].[LaborHoursVariance],

[v\_LaborHoursVariance].[PercentHoursVariance]

FROM

[v\_LaborHoursVariance]

INNER JOIN [v\_JobManagerNames] ON

[v\_LaborHoursVariance].[JobID] = [v\_JobManagerNames].[JobID]

WHERE

[v\_LaborHoursVariance].[PercentHoursVariance] =

(SELECT

MAX([v\_LaborHoursVariance].[PercentHoursVariance])

FROM

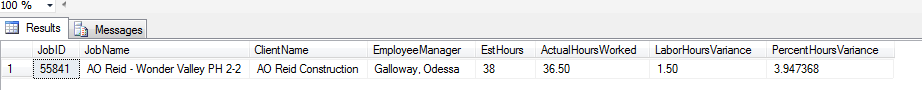
[v\_LaborHoursVariance]

)

ORDER BY

[v\_LaborHoursVariance].[JobID]

;



/\*\*

\* Exercise 07

\*

\* What is the average amount of time (labor hours) spent on a

\* \underline{completed} job task per square foot, as compared to the

\* estimated amount of time that should be spent on a task per square foot?

\*

\* Use the data in the JobTask table to calculate the average amount of

\* EstHours/Squarefeet, but use the data in the TimeSheet table to calculate

\* the average amount of time that was actually worked on a completed task.

\* I recommend creating separate views for the estimated hours per square

\* feet and the actual hours per square feet. The estimate view is a little

\* easier to create because it doesn’t require a join. Include all rows in the

\* JobTask table to get the average EstHours/Squarefeet for a task. To get the

\* average \underline{actual} hours per square feet requires that you join the

\* TimeSheet table and the JobTask table to be able to use the square feet in

\* the JobTask table. Do not include data for incompleted tasks when

\* calculating the ActualHours/SquareFeet. Remember that you have to SUM the

\* HoursWorked in the TimeSheet table by JobID and TaskID to get the Actual

\* HoursWorked from the TimeSheet table. I rounded the final results to 6

\* digits after the decimal point. The result table is at the top of the

\* next page. There is one row in the result table for each row in the Task

\* table. Sort the result table by TaskID.

\*

\* The ComparisonMessage should be generated as shown on the result table

\* above; if both the EstimatedHours and ActualHours are null, then put the

\* message ``Null Estimate" in the ComparisonMessage column. Remember that

\* a CASE statement in the SELECT list executes sequentially, so whatever

\* WHEN statement is placed first will be executed first. The CASE

\* statement stops executing as soon as a WHEN condition is true.

\* Potential problem: EstHours and Squarefeet are integers and must

\* be converted to decimal data types before they can be used in a

\* calculation that could generate a decimal result.

\*/

DROP VIEW [v\_EstHoursPerSqFt];

CREATE VIEW [v\_EstHoursPerSqFt] AS

SELECT

[JobTask].[TaskID] AS 'TaskID',

SUM([JobTask].[EstHours]) /

CAST(SUM([JobTask].[SquareFeet]) AS DECIMAL(6, 2)) AS 'EstimatedHoursPerSqFt'

FROM

[JobTask]

GROUP BY

[JobTask].[TaskID]

;

----------

CREATE VIEW v\_CompleteSqFtSum AS

SELECT

TaskID,

SUM(SquareFeet) 'SquareFeet'

FROM

JobTask

WHERE

DateCompleted IS NOT NULL

GROUP BY

TaskID

;

CREATE VIEW v\_CompleteActualHours AS

SELECT

TimeSheet.TaskID,

SUM(TimeSheet.HoursWorked) 'Hours'

FROM

TimeSheet

INNER JOIN JobTask ON

TimeSheet.JobID = JobTask.JobID AND TimeSheet.TaskID = JobTask.TaskID

WHERE

JobTask.DateCompleted IS NOT NULL

GROUP BY

TimeSheet.TaskID

;

CREATE VIEW v\_CompleteActualHoursPerSqFt AS

SELECT

A.TaskID,

A.[Hours] / S.SquareFeet 'ActualHoursPerSqFt'

FROM

v\_CompleteActualHours A

INNER JOIN v\_CompleteSqFtSum S ON

A.TaskID = S.TaskID

;

DROP VIEW [v\_ActualHoursPerSqFt];

CREATE VIEW [v\_ActualHoursPerSqFt] AS

SELECT

[TimeSheet].[TaskID] AS 'TaskID',

SUM([TimeSheet].[HoursWorked]) /

SUM([JobTask].[SquareFeet]) AS 'ActualHoursPerSqFt'

FROM

[v\_TimeSheetsWithPay] [TimeSheet]

INNER JOIN [JobTask] ON

[TimeSheet].[JobID] = [JobTask].[JobID] AND

[TimeSheet].[TaskID] = [JobTask].[TaskID]

WHERE

[JobTask].[DateCompleted] IS NOT NULL

GROUP BY

[TimeSheet].[TaskID]

;

------------------------------

DROP VIEW [v\_HoursPerSqFtComparison];

CREATE VIEW [v\_HoursPerSqFtComparison] AS

SELECT

[Task].[TaskID] AS 'TaskID',

[Task].[TaskDescription] AS 'TaskDescription',

[v\_EstHoursPerSqFt].[EstimatedHoursPerSqFt] AS 'EstimatedHoursPerSqFt',

[v\_CompleteActualHoursPerSqFt].[ActualHoursPerSqFt] AS 'ActualHoursPerSqFt',

CASE

WHEN [v\_EstHoursPerSqFt].[EstimatedHoursPerSqFt] IS NULL

THEN 'Null Estimate'

WHEN [v\_CompleteActualHoursPerSqFt].[ActualHoursPerSqFt] IS NULL

THEN 'Null Actual'

WHEN [v\_EstHoursPerSqFt].[EstimatedHoursPerSqFt] > [v\_CompleteActualHoursPerSqFt].[ActualHoursPerSqFt]

THEN 'Estimate Larger'

WHEN [v\_CompleteActualHoursPerSqFt].[ActualHoursPerSqFt] > [v\_EstHoursPerSqFt].[EstimatedHoursPerSqFt]

THEN 'Actual Larger'

ELSE

'No Difference'

END /\*\*/ AS 'Comparison Message'

FROM

[Task]

LEFT OUTER JOIN [v\_EstHoursPerSqFt] ON

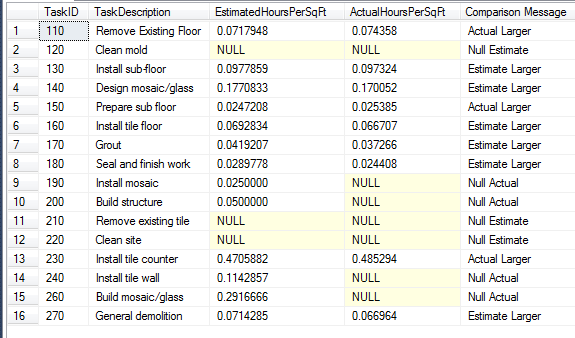
[Task].[TaskID] = [v\_EstHoursPerSqFt].[TaskID]

LEFT OUTER JOIN [v\_CompleteActualHoursPerSqFt] ON

[Task].[TaskID] = [v\_CompleteActualHoursPerSqFt].[TaskID]

;

SELECT \* FROM v\_HoursPerSqFtComparison;



/\*\*

\* Exercise 08

\*

\* Use the result table generated for question #7 to help you answer

\* this question. The goal of this query is to identify which task has

\* the largest negative difference between the EstimatedHoursPerSqFt and

\* ActualHoursPerSqFt (which estimate is the worst because the actual

\* is larger).

\*/

SELECT

\*

FROM

v\_HoursPerSqFtComparison Comparison

WHERE

Comparison.EstimatedHoursPerSqFt - Comparison.ActualHoursPerSqFt = (

SELECT

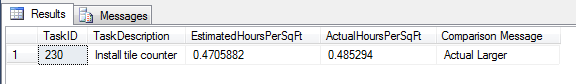
MIN(EstimatedHoursPerSqFt - ActualHoursPerSqFt)

FROM

v\_HoursPerSqFtComparison

)

;



/\*\*

\* Exercise 09

\*

\* The objective of this query is similar to that for question #7, except

\* this time we are going to look at labor costs rather than labor hours.

\* What is the average estimated labor cost per square foot as compared

\* to the actual labor cost per square foot for each task? I recommend

\* looking back at question #1, where you probably created a view to help

\* you calculate actual labor costs for a task on a job. That view will help

\* you with this question. Do \textbf{\underline{not}} include data for

\* incompleted tasks when calculating the actual labor cost/SquareFeet;

\* do include data for incompleted tasks when calculating the estimated

\* labor cost/squarefeet.

\*/

DROP VIEW [v\_EstWagesPerSqFt];

CREATE VIEW [v\_EstWagesPerSqFt] AS

SELECT

[JobTask].[TaskID] AS 'TaskID',

CAST(SUM([JobTask].[EstLaborCost]) AS DECIMAL) /

CAST(SUM([JobTask].[SquareFeet]) AS DECIMAL(6, 2)) AS 'EstimatedWagesPerSqFt'

FROM

[JobTask]

GROUP BY

[JobTask].[TaskID]

;

----------

CREATE VIEW v\_ActualPay AS

SELECT

TS.TaskID,

SUM(TS.LaborCost) 'ActualWagesPerSqFt'

FROM

v\_TimeSheetsWithPay TS

INNER JOIN JobTask JT ON

TS.JobID = JT.JobID AND TS.TaskID = JT.TaskID

WHERE

JT.DateCompleted IS NOT NULL

GROUP BY

TS.TaskID

;

DROP VIEW [v\_ActualWagesPerSqFt];

CREATE VIEW [v\_ActualWagesPerSqFt] AS

SELECT

[v\_ActualPay].[TaskID] AS 'TaskID',

v\_ActualPay.ActualWagesPerSqFt /

v\_CompleteSqFtSum.SquareFeet AS 'ActualWagesPerSqFt'

FROM

[v\_ActualPay]

INNER JOIN [v\_CompleteSqFtSum] ON

[v\_ActualPay].[TaskID] = [v\_CompleteSqFtSum].[TaskID]

;

------------------------------

SELECT

[Task].[TaskID] AS 'TaskID',

[Task].[TaskDescription] AS 'TaskDescription',

[v\_EstWagesPerSqFt].[EstimatedWagesPerSqFt] AS 'EstimatedLaborCostPerSqFt',

[v\_ActualWagesPerSqFt].[ActualWagesPerSqFt] AS 'ActualLaborCostPerSqFt',

CASE

WHEN [v\_EstWagesPerSqFt].[EstimatedWagesPerSqFt] IS NULL

THEN 'Null Estimate'

WHEN [v\_ActualWagesPerSqFt].[ActualWagesPerSqFt] IS NULL

THEN 'Null Actual'

WHEN [v\_EstWagesPerSqFt].[EstimatedWagesPerSqFt] > [v\_ActualWagesPerSqFt].[ActualWagesPerSqFt]

THEN 'Estimate Larger'

WHEN [v\_ActualWagesPerSqFt].[ActualWagesPerSqFt] > [v\_EstWagesPerSqFt].[EstimatedWagesPerSqFt]

THEN 'Actual Larger'

ELSE

'No Difference'

END /\*\*/ AS 'Comparison Message'

FROM

[Task]

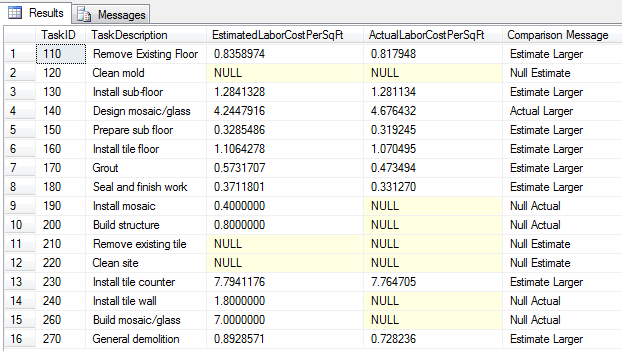
LEFT OUTER JOIN [v\_EstWagesPerSqFt] ON

[Task].[TaskID] = [v\_EstWagesPerSqFt].[TaskID]

LEFT OUTER JOIN [v\_ActualWagesPerSqFt] ON

[Task].[TaskID] = [v\_ActualWagesPerSqFt].[TaskID]

;



/\*\*

\* Exercise 10

\*

\* Which clients did not have any jobs with a DateAccepted last year?

\* Which materials were not assigned (DateAssigned) to any job tasks last year?

\* Combine the clients and materials into a single result table (hint: Use the

\* UNION statement). Make sure that you use the GETDATE() function to

\* determine the correct year.

\*/

DROP VIEW [v\_ClientsWithoutJobsInPreviousYear];

CREATE VIEW [v\_ClientsWithoutJobsInPreviousYear] AS

SELECT

[Client].[ClientID] AS 'ClientID',

[Client].[ClientName] AS 'ClientName'

FROM

[Client]

WHERE

[Client].[ClientID] NOT IN (

SELECT

ClientID

FROM

Job

WHERE

DATEDIFF(YEAR, DateAccepted, GETDATE()) = 1

)

;

DROP VIEW v\_MaterialsAssignedYearPrevious;

CREATE VIEW v\_MaterialsAssignedYearPrevious AS

SELECT

DISTINCT(Material.MaterialID)

FROM

MaterialAssigned

INNER JOIN MaterialPurchased ON

MaterialAssigned.POID = MaterialPurchased.POID

INNER JOIN Material ON

MaterialPurchased.MaterialID = Material.MaterialID

WHERE

DATEDIFF(YEAR, MaterialAssigned.DateAssigned, GETDATE()) = 1

;

DROP VIEW v\_MaterialsNotAssignedYearPrevious;

CREATE VIEW v\_MaterialsNotAssignedYearPrevious AS

SELECT

MaterialID,

MaterialName

FROM

Material

WHERE

MAterial.MaterialID NOT IN (SELECT \* FROM v\_MaterialsAssignedYearPrevious)

SELECT

'Client: ' + CAST(ClientID AS VARCHAR) AS 'ClientOrMaterialID',

'Client: ' + ClientName AS 'ClientOrMaterialName'

FROM

v\_ClientsWithoutJobsInPreviousYear

/\*\*/

UNION

/\*\*/

SELECT

'Material: ' + CAST(MaterialID AS VARCHAR) AS 'ClientOrMaterialID',

'Material: ' + MaterialName AS 'ClientOrMaterialName'

FROM

v\_MaterialsNotAssignedYearPrevious

;

