select current\_timestamp

* Get plan using query menu, toolbar, or Ctrl+L
* Query number
* query cost
* query statement
* graphical description of plan
* nodes are discrete operations
* node icon, name, cost

select \* from Person.Person

* two nodes
* connecting arrows show data flow
* node on left receive data from node(s) on right (usually one or two)
* usually read plans from right to left
* each node has its own (likely different) cost
* clustered index scan includes object name
* discuss what is a clustered index

select \* from Person.Person

select \* from Person.EmailAddress

* cost on queries is relative to the batch
* cost on nodes is relative to the query
* relevant to stored procedures

select \* from Person.Person

* hover over nodes and arrow
* right-click on plan background, show menu options
* view XML
* right-click nodes, show properties
* briefly discuss parallelism after showing SELECT properties

**Discuss cost here**

select \* from Sales.SalesOrderDetail

* additional steps for computed column
* show how to see computation in properties

select \* from Person.Person

where BusinessEntityID = 33

* Show seek instead of scan
* Hover and show seek predicate
* Discuss seek vs. scan
* See if they notice the change in the arrow
* Show properties for SELECT and point out parameter sniffing

**Show actual execution plan for:**

select \* from Person.Person

* show differences in graph, hover, and properties
* **Actual I/O Statistics**, logical vs. physical reads
* discuss reads (pages) vs. rows
* clear cache, read with predicate
* show actual reads
* read w/out predicate
* show logical vs. physical reads (**3821**)
* discuss when actual is different from estimated (statistics)

select \* from Person.Person p

left join Person.EmailAddress a on a.BusinessEntityID = p.BusinessEntityID

* show different costs for different nodes
* discuss merge join
* two lines going into it
* top table (Person) is outer / primary table
* logical vs physical operation
* subtree cost (operation cost + cost of each child node)
* merge / where columns in hover and properties

select \* from Person.Person

where LastName = 'Smith' and Title = 'Mr.'

* plan looks like a join, but there is only one table
* show more data (arrow width) from index than from table
* look at index seek
* show very low cost, seek predicate, and output list in hover
* note number of rows (103)
* look at key lookup properties
* note number of executions
* note number of rows
* note operator cost vs. CPU and I/O cost (cost \* executions)
* note output list does not include columns from index
* note object is PK and seek predicate on BusinessEntityId
* explain how columns from PK are added to columns from index
* note predicate on Title
* look at description and outer reference in nested loop

insert into Person.Person...

* note insert into PK and other normal indexes on top line
* note check for BusinessEntity.BusinessEntityId
* note other streams are inserting into XML indexes

update Person.PersonPhone...

* note lookup of PersonPhone by BusinessEntityId on right end of top line
* note check for PhoneNumberType.PhoneNumberTypeID

delete from Person.BusinessEntity...

* note delete operation on right end of top line
* note checks in other tables for FK references

select \* from HumanResources.vEmployee

* note fairly complex plan
* note RetrievedFromCache property of SELECT is false
* get estimated plan again and note it is still false
* execute query, then get estimated plan and note it is now true
* discuss potential problem with bad cached query
* clear proc cache
* get estimated plan again and note it is false again

select \* from Person.Person p

left join Person.EmailAddress2 a on a.BusinessEntityID = p.BusinessEntityID

where LastName = 'Smith'

* discuss missing index hint
* right-click, missing index details
* discuss why not blindly follow the advice