****

**VitalCenter Online**

*SQL Workshop*

*Practicum One*

# Background

## **Objectives and Assumptions**

Objectives:

* Introduce staff to a base line of SQL skills so they can review a standard extract and understand what is going on

Prerequisites:

* Users have access to SSMS on a machine that is in the Galen network
* Users have access to one of the Virtual servers: <https://testvms.galenhealthcare.com/>

## Reference:

**For basic SQL Syntax reference, use these as a starting point:**

www.w3schools.com/sql/default.asp

www.sqlcourse.com

intranet.galenhealthcare.com/index.php?title=SQL\_Scripting\_-\_General

intranet.galenhealthcare.com/index.php?title=Introduction\_to\_SQL

intranet.galenhealthcare.com/index.php?title=SQL\_Coding\_Guidelines

|  |  |
| --- | --- |
| Leasing the Virtual Machine **Navigate to the VM Manager** testvms.galenhealthcare.com  **Enter the number of hours for the machine you’ve been assigned.**  **Click “Lease.”**  **Galen incurs costs while these are running, so when they are not in use, “stop” the VM!** |  |
| Connect to VM & Works DB **Open SSMS and verify connectivity.**  **Open SSMS and create a connection to the appropriate VM.**  (Once it has moved into “Running” status it can still take upwards of 1-2 minutes before actually connected.)  **Expand (+) the Works database**.  SSMS can be used on the VM desktop, but if connection speeds are slow, SSMS can be used on your local machine as well, if your machine is on a Galen office network or VPN. Connect with the same credentials as here to test.  . |  |
| IntelliSense Enabled **Verify that SSMS IntelliSense is enabled. This will enable spell check, error check, and auto-fill capabilities of SSMS to speed up your work.** |  |

# Selecting Data

|  |  |
| --- | --- |
| *Select the correct database*  **When first connected to a server, the “master” db will default in, and queries will show an error unless they are fully qualified.**  **Change the db from “master” to “Works” in the drop down and notice the change**  A fully qualifed name has three parts: **database .** **schema** . **table**  A “..” can be substituted for “dbo” |  |
| **Run the following scripts** : | SELECT \* FROM Works.dbo.Person  SELECT FirstName, lastname, DateOfBirth FROM Works.dbo.Person  SELECT FirstName, lastname, DateOfBirth, \* FROM Works.dbo.Person  **How many rows were returned in each?**  **How long did it take to compile each?**  **How many fields were returned in each?**  SELECT LastName, FirstName FROM Works.dbo.Person Order by LastName Asc  SELECT \* FROM Works.dbo.Person Order by DateOfBirth Desc  SELECT LastName, FirstName, SexDE, SSN FROM Works.dbo.Person Order by SexDE, SSN Desc  **If not specified, what is the default for sort order?** |

# Limiting Data

|  |  |
| --- | --- |
| **Run the following scripts which use the** Where **clause and a standard set of qualifiers:**  Make sure each field listed gets used at least once.  Replace each **???** with an appropriate value(s)   * =, <, >, <>, <=, >= * IN, NOT IN * AND, OR * LIKE, NOT LIKE | SELECT LastName, FirstName, DateOfBirth, SSN, SexDE From dbo.Person Where {field} = '???'  SELECT LastName, FirstName, DateOfBirth, SSN, SexDE From dbo.Person Where {field } < '???'  SELECT LastName, FirstName, DateOfBirth, SSN, SexDE From dbo.Person Where {field} > '???'  SELECT LastName, FirstName, DateOfBirth, SSN, SexDE From dbo.Person Where {field} <> '???'  SELECT LastName, FirstName, DateOfBirth, SSN, SexDE From dbo.Person Where {field} IN ('???','???')  SELECT LastName, FirstName, DateOfBirth, SSN, SexDE From dbo.Person Where ({field } = '???' AND { field } = '???') OR {field}= '???' -- Mind the ()  SELECT LastName, FirstName, DateOfBirth, SSN, SexDE From dbo.Person Where {field} like '%???%' --Anywhere  SELECT LastName, FirstName, DateOfBirth, SSN, SexDE From dbo.Person Where {field} like '%???' --Ending  SELECT LastName, FirstName, DateOfBirth, SSN, SexDE From dbo.Person Where {field} like '???%' --Beginning |
| **Contrast/Compare the following queries.**   * 1. Right click the Person table and click “Select Top 100 Rows”   2. SELECT TOP 100 \*   FROM dbo.Person  **When might either one be more helpful?** |  |

Aliasing

|  |  |
| --- | --- |
| **Run & Review the following:**  **Add three more fields to each and give each an appropriate ALIAS.**  **What is going on with the Status field?** | Select d.DocumentId as [Document Id]  , d.PatientId as 'Pat id'  , d.RecordedDTTM as "Recorded DTTM"  , d.[Status] as Status  ,\*  From Works..Document as d  SELECT p.FirstName as FName  ,p.lastname as LName  ,p.DateOfBirth as DOB  FROM Works..Person as p  ORDER BY DOB |

# String Functions

|  |  |
| --- | --- |
| **Run & Review the following:**   * Upper / Lower * Left / Right * Len | Select UPPER(LastName), LOWER(FirstName) from Person  Select LastName, Left(FirstName,1) as First\_Init from Person  Select LastName, FirstName, SSN, Len(SSN) as SSN\_Length from Person Where Len(SSN) < 9 |
| * LTrim / RTrim   **Copy each SSN output value into Notepad++ and set “View>Show Symbol>Show All Characters” and compare results.** | Select LastName, FirstName, SSN, Len(SSN) as SSN\_Length, LTrim(SSN) as LSSN, RTrim(SSN) as RSSN, LTrim(RTrim(SSN)) as LRSSN from Person |
| * Concat   **Compare and contrast the output.**  **What happens with NULL data?** | Select Concat(last\_nm, ', ', Mother\_Maiden\_nm) as TestName, \* from Patient  Select last\_nm + Mother\_Maiden\_nm as TestName, \* from Patient  Select last\_nm + first\_nm as TestName, \* from Patient  Select Concat(last\_nm, ', ' + Mother\_Maiden\_nm) as TestName, \* from Patient |
| * Case   **When is either method appropriate?** | Select LastName, Case When SSN = '' Then 'N/A' Else SSN End as SSN from Person  Select LastName,  Case When SSN = '' Then 'N/A'  When SSN = 'NA' Then 'N/A'  When SSN Is Null Then 'N/A'  When DateOfBirth = '' Then 'N/A'  End as SSN  From Person  Select LastName, Case IsInactiveFLAG  When 'N' Then 'NO'  When 'Y' then 'YES'  Else 'N/A'  End as Inactive from Person |
| * IsNull / Coalesce   **Compare and contrast the output. What happens with NULL data? When is either method appropriate?** | Select LastName, FirstName, IsNull(DateOfBirth,'') as DOB from Person  Select ItemID, Coalesce(DateToAdminister, ExpectedActionDTTM ,ExpirationDT) as DateValue from Medication  Select ItemID, Coalesce(DateToAdminister,ExpirationDT) as DateValue from Medication  Select ItemID, Coalesce(DateToAdminister,ExpirationDT,'') as DateValue from Medication  Select VisitID, EncounterID, IsNull(OrderActivityHeaderID,0) as TestID from Charge  Select VisitID, EncounterID, Coalesce(OrderActivityHeaderID,EncounterID) as TestID from Charge |
| NullIf **How many Documents are returned?** | When searching for empty value, with fields that are NULLABLE both blank and NULL values should be searched for.  Select p.LastName, p.FirstName From Person as P  Join Document as d on d.PatientID = p.ID  Where d.AccessionNumber Is Null  Select p.LastName, p.FirstName From Person as P  Join Document as d on d.PatientID = p.ID  Where d.AccessionNumber = ''  Select p.LastName, p.FirstName From Person as P  Join Document as d on d.PatientID = p.ID  Where d.AccessionNumber = '' or d.AccessionNumber Is Null  /\* NullIf Solution \*/  Select p.LastName, p.FirstName From Person as P  Join Document as d on d.PatientID = p.ID  Where NullIf(d.AccessionNumber,'') Is Null |

# Aggragate Functions

|  |  |
| --- | --- |
| Distinct & Group By **Run & Review the following:** | Select Distinct LastName from Person  Select LastName from Person Group by LastName |
| Count() **Run & Review the following:**  **Note: Nulls will not be counted! Use (\*) instead of 1 or field.** | Select Count(\*) as Rows, 'Works Person' as [Table] from Person  Union  Select Count(\*), 'Works Provider' from Provider  Union  Select Count(\*), 'AETL Person' from AETL\_PHIL.Customer.Person  Union  Select Count(IsSourceSystemUser), 'AETL SourceSystemUser' from AETL\_PHIL.Customer.Person  Order by 2 Desc -- column order only available in Order By |
| Min() & Max() **Run & Review the following:** | Select p.LastName, p.FirstName, Max(DTTM) as MaxEncounterDTTM  From Person as p  Join Encounter as e on e.patientID = p.id  Where P.id = 30  Group by p.LastName, p.FirstName |

# Window Functions

|  |  |
| --- | --- |
| Row\_Number() **Run & Review the following:**  **Run the subquery in the second query alone. What is it doing?**  **Run the second data set to not return the first alphabetical entry**  Cannot be used when UPDATING rows | Select  Row\_Number() Over (Order by LastName, FirstName) as NumberedList  ,Row\_Number() Over (Partition by LastName Order by LastName, FirstName) as NumberedListByName  ,LastName  ,FirstName  From Person  -- Find first instance of each LastName  Select \*  From Person as P  Join (Select ID, Row\_Number() Over (Partition by LastName Order by LastName, FirstName) as RowNum, LastName From Person) as Top1 on Top1.ID = P.Id  Where Top1.RowNum = 1 |
| Rank() & Dense\_Rank() **Run & Review the following:**  **Note the difference here possibily**  **Which Rank value is most useful in this instance?** | Select  Row\_Number() Over(Order by LastName) as RowNumber  ,Row\_Number() Over (Partition by LastName Order by LastName, FirstName) as Partitioned  ,Rank() Over (Order by LastName) as Ranked  ,Dense\_Rank() Over ( Order by LastName) as DenseRank  ,LastName  ,FirstName  From Person |

# Joins

Add cross and outer applies

|  |  |
| --- | --- |
| Inner Join / Join **Run & Review the following:** | SELECT p.ID as Person\_ID, p.LastName, p.FirstName, pr.NPI as Provider\_NPI  FROM Person as p  **Inner Join** Provider as pr ON pr.ID = p.ID  Where pr.IsPCPFlag = 'N' |
| Left Join **Run & Review the following:**  **What happens when you add “Where pr.IsPCPFlag IS NULL”**  **to the first query?**  **Compare output when adding a WHERE using Table B. Why are there missing rows?** | SELECT p.ID as Person\_ID, p.LastName, p.FirstName, pr.NPI as Provider\_NPI  FROM Person as p  **Left Join** Provider as pr ON pr.ID = p.ID  SELECT p.ID as Person\_ID, p.LastName, p.FirstName, pr.NPI as Provider\_NPI  FROM Person as p  **Left Join** Provider as pr ON pr.ID = p.ID  Where pr.IsPCPFlag = 'N'  SELECT p.ID as Person\_ID, p.LastName, p.FirstName, pr.NPI as Provider\_NPI  FROM Person as p  Left Join Provider as pr ON pr.ID = p.ID and pr.IsPCPFlag = 'N' |
| Multi Table Join **Run & Review the following:**  **Reset with table that has NULL instead of 0 for subtables** | SELECT p.ID, p.LastName, p.FirstName, pr.NPI as Provider\_NPI, s.EntryName as Specialty, rd.EntryName as Resource, dt.EntryName -- ,s.\*  FROM Provider as pr  Inner Join Person as p ON p.ID = pr.ID  Left Join Specialty\_DE as s ON s.ID = pr.SpecialtyDE  Left Join Resource\_DE as rd on rd.ID = pr.ResourceDE  Left Join Distribution\_Type\_DE as dt on dt.ID = pr.DistributionTypeDE  Order by p.ID |

|  |  |
| --- | --- |
| **Run the following script** {adapted from a standard VCO Allergy query] Tables can be linked to tables at any level  **Why was Allergen\_Reaction\_DE joined numerous times?**  **\*BEST PRACTICE\***   * Notice Column name alias and table join tabbed spacing for more rapid reference (not all need be included) when viewed in **SSMS**. * Notice use of (NOLOCK) in table joins. * Notice use of 1=1 in Where * Notice Indenting structure | Select  ia.ID as ExternalDataId,  a.ID as ExternalDataVersion,  ia.PatientID as [PatientExternalDataId],  IsNull(e.DTTM,'1901-01-01') as ClinicallyRelevantDttm,  Coalesce(rtrim(mad.EntryName), rtrim(ae.EntryName),'Unknown') as TypeName,  e.DTTM as ServiceDttm,  ast.EntryName as Status,  Case when ast.EntryCode = '3' then 1 end as [IsInvalidated],  Case when ast.EntryCode = '3' then a.RecordedDTTM end as InvalidatedDttm,  case when ast.EntryCode = '3' then convert(varchar,a.WhoDidItID) end as InvalidatedByProviderExternalDataId,  case when mad.EntryCode is null then 0 else 1 end as IsMedication,  STUFF(coalesce(';' + NULLIF(rtrim(a.ReactionDET),''), isnull(';' + nullif(rtrim(r1.EntryName),''),'') + isnull(';' + nullif(rtrim(r2.EntryName),''),'') + isnull(';' + nullif(rtrim(r3.EntryName),''),'')  ),1,1,'') as ReactionDescription,  a.RecordedDTTM as RecordedDttm,  convert(datetime, ia.CreateDTTM) as ExternalDataCreatedDttm,  convert(datetime, ia.LastUpdateDTTM) as ExternalDataUpdatedDttm,  convert(varchar,a.WhoDidItID) as PerformingProviderExternalDataId,  coalesce(rtrim(mad.EntryName), rtrim(ae.EntryName),'Unknown') as Title,  rtrim(ast.EntryName) as VcoSetEntries,  aha.EncounterID as EncounterExternalDataId  From Allergy as a (NOLOCK)  Inner Join Item\_Allergy as ia (NOLOCK) on ia.ID = a.ItemID  Inner Join Act\_Hdr\_Allergy as aha (NOLOCK) on aha.CurrentID = ia.CurrentID  Inner Join encounter as e (NOLOCK) on e.ID = aha.EncounterID  Left Join Medication\_allergy\_de as mad (NOLOCK) on mad.ID = a.MedAllergyDE and a.MedAllergyDE <> 0  Left Join medication\_de as med (NOLOCK) on med.id = mad.meddictde Left Join allergen\_de as ae (NOLOCK) on ae.id = a.AllergenDE and a.AllergenDE not in (0,38)  Left Join allergen\_reaction\_de as r1 (NOLOCK) on r1.id = a.reaction1de  Left Join allergen\_reaction\_de as r2 (NOLOCK) on r2.id = a.reaction2de  Left Join allergen\_reaction\_de as r3 (NOLOCK) on r3.id = a.reaction3de  Left Join allergy\_status\_de as ast (NOLOCK) on ast.ID = a.AllergyStatusDE  Where 1=1  and aha.PatientID <> 0 |
| **Run & Review the following:**  **Adapt the second query to pull in usable values**  Many times, a single table will hold mainly **ID** values and so other tables with meaningful text values must be linked in to retrieve a “name**.”** | Select a.ID, a.LocationDe, a.ResourceDE, a.AppointmentTypeDe, a.AppointmentStatusDE, a.OrganizationID From Appointment as a Order By A.ID  Select a.ID as Appt\_ID  ,a.LocationDe  ,l.EntryName as locationName  ,a.ResourceDE  ,r.EntryName as ResourceName  ,a.AppointmentTypeDe  ,atd.EntryName  ,a.AppointmentStatusDE  ,asd.EntryName  ,a.OrganizationID  ,o.Name  From Appointment as a  Join Location\_DE as l on l**.???** = a**.???**  Join Resource\_DE as r on r**.???** = a**.???**  Join Appointment\_Type\_DE as atd on atd.**???**= a**.???**  Join Appointment\_Status\_DE as asd on asd**. ???** = a**.???**  Join Organization as o on o. **???** = a.**???**  Order By A.ID |

# Converting Data

|  |  |
| --- | --- |
| Cast & Convert **Run & Review the following field conversions**  **Run with the first commented line and then switch with the second and then with the third**. | SELECT  PerformedDTTM  ,Cast(PerformedDTTM as Date) as Cast1  ,Cast(PerformedDTTM as Time) as Cast2  ,Convert(varchar(16),PerformedDTTM,110) as Convert1  ,Convert(DateTime,PerformedDTTM,110) as Convert2  ,Convert(varchar,PerformedDTTM,120) as Convert3  ,Convert(varchar(MAX),itemtext) as Convert4  --,ItemId + ItemText + recordeddttm as Concat1  --,Cast(ItemId as VarChar) + ItemText + Cast(recordeddttm as VarChar) as Concat2  --,Concat(ItemId, ItemText, recordeddttm) as Concat3  FROM item\_annotation IA2 (NOLOCK)  Convert(,,) DateTIme style reference:  <https://docs.microsoft.com/en-us/sql/t-sql/functions/cast-and-convert-transact-sql> |
| Replace **Run & Review the following:**  <https://theasciicode.com.ar/ascii-control-characters/null-character-ascii-code-0.html>  **~~Note: \.br\ is an Epic standard value for a line break~~** | Select  Replace (Replace (Replace (Replace(rt.resulttext,'|',''), Char(13)+Char(10),'\.br\'), Char(13),'\.br\'), Char(10),'\.br\')  ,Replace (Replace (Replace (r.answerdet, Char(13)+Char(10), '\.1br\'), Char(13), '\.2br\'), Char(10),'\.3br\')  From Item\_Result ir  Join Result r ON r.ID = ir.CurrentID  Left Join Result\_Text rt ON rt.ResultID = r.ID  Join QO\_Classification\_DE qcd ON qcd.ID = ir.QOClassificationDE  Where 1=1  --and (r.answerdet like '%' + Char(10) + '%' or r.answerdet like '%' + Char(13) + '%') |
| Update **Run & Review the following:**  Try using the table Alias for the field Misc3 | Update AETL\_Touchworks.customer.Person  Set Misc1 = 'Employer' , Misc2 = 'Medical IT'  Where LastName = 'Allscripts'  Update AETL\_Touchworks.customer.Person  Set AETL\_Touchworks.customer.Person.Misc3 = pp.Number  From AETL\_Touchworks.customer.Person as p  Join AETL\_Touchworks.customer.PersonPhone as pp on pp.ExternalDataId = p.ExternalDataId  Where pp.IsPrimary = 1 |

# Using Ad Hoc Tables

|  |  |
| --- | --- |
| SubQuery (Where clause) | Select p.ID, p.LastName, p.FirstName,SSN, DateofBirth, LastEditedDttm  From Person as p  Where P.ID in (Select ID from Provider)  Order by p.ID |
| Derived (From clause) | Select p.ID, p.LastName, p.FirstName,SSN, DateofBirth, LastEditedDttm  From Person as p  Join (Select ID from Provider) as pr on pr.id = p.id  Order by p.ID |
| Common Table Expression **Run the contents of the CTE script alone and note the outcome.**  **Run the entire scripts and note the outcome.**  **After running the scripts try to query the TABLE you created. Does it work?**  This should be used for smaller data sets as it can’t be indexed and can’t be used in subqueries | /\*\*\*\*\* The CTE starts here \*\*\*\*\*/  With ItemAnnotation as (  select ItemID,  stuff((SELECT char(13) + char(10) + replace(convert(varchar,PerformedDTTM,106),' ','') + replace(' ' + right(convert(varchar,PerformedDTTM),7),' ',' ') + ' by ' + rtrim(works.dbo.fnGSelect etPersonFullName(IA2.AnnotatedByID)) + ': ' + convert(varchar(MAX),itemtext)  FROM works.dbo.item\_annotation IA2 (NOLOCK)  where IA2.ItemID = IA.ItemID  and IA2.itemtype = IA.itemtype  order by PerformedDTTM DESC  FOR XML PATH(''), type  ).value('.', 'varchar(max)'),1,2,''  ) as Annotation  from works.dbo.item\_annotation IA (NOLOCK)  where IA.itemtype = 'AL'  group by IA.ItemID, IA.itemtype  )  /\*\*\*\*\* The CTE ends here  SELECT  ia.ID as [ExternalDataId]  ,a.ID as [ExternalDataVersion]  ,ROW\_NUMBER ( ) OVER (PARTITION BY ia.ID ORDER BY a.ID asc) as [DisplayOrder]  ,e.DTTM as [ClinicallyRelevantDttm]  ,Coalesce(mad.EntryCode,ae.EntryCode) as [TypeCode]  ,Coalesce(mad.EntryName,ae.EntryName) as [TypeName]  ,IAN.Annotation as [Comments]  From works.dbo.Allergy a (NOLOCK)  Inner Join Item\_Allergy as ia (NOLOCK) on ia.ID = a.ItemID  Left Join act\_allergy as aa (NOLOCK) on aa.ActivityHeaderID= ia.ActivityHeaderID  Inner Join Act\_Hdr\_Allergy as aha (NOLOCK) on aha.CurrentID = ia.CurrentID  Inner Join encounter as e (NOLOCK) on e.ID = aha.EncounterID  Left Join Medication\_allergy\_de as mad (NOLOCK) on mad.ID = a.MedAllergyDE and a.MedAllergyDE <> 0  Left Join allergen\_de as ae (NOLOCK) on ae.id = a.AllergenDE and a.AllergenDE not in (0,38)  Left Join allergy\_status\_de as ast (NOLOCK) on ast.ID = a.AllergyStatusDE  Left Join ItemAnnotation as IAN on IAN.ItemID = ia.ID  Where aha.PatientID = 19 |

|  |  |
| --- | --- |
| Temp Tables **Run the following scripts and note the outcomes**  Unline a CTE, A Temp Table will last as long as the connection to it (session/window) exists. They must be removed before then can be recreated.  Note the location of #TempPerson    A **Global** temporary table is visible to all other sessions. Global temp tables are only destroyed when the current user disconnected and all the sessions referring to it closed.  **Also maybe “create” temp table and add an index.** | Use Works  Go  Select ID, LastName, Firstname  Into #TempPerson  From Person;  Select \* From #TempPerson  --  Select \* From #TempPerson  Select \* From #TempPerson -- Run this one line in a new tab  --  Select ID, LastName, Firstname Into #TempPerson From Person  --  Select \* From #TempPerson –- run multiple times to see is still there  --  Select ID, LastName, Firstname Into #TempPerson From Person  --  Drop Table #TempPerson  --  Select \* From #TempPerson  /\*\*\*\*\* Include DROP TABLE \*\*\*\*\*/  DROP TABLE IF EXISTS #TempPerson;  Select ID, LastName, Firstname  Into #TempPerson  From Person;  Select \* From #TempPerson  /\*\*\*\*\* “CREATE” Temp Table \*\*\*\*\*/  CREATE TABLE #name\_of\_temp\_table (  column\_1 datatype,  column\_2 datatype,  column\_3 datatype,  column\_n datatype )  INSERT INTO #name\_of\_temp\_table (column\_1, column\_2, column\_3)  SELECT column\_1, column\_2, column\_3  FROM table\_name  WHERE condition  /\*\*\*\*\* Global Temp Table \*\*\*\*\*/  DROP TABLE IF EXISTS ##GlobalTempPerson;  Select ID, LastName, Firstname  Into ##GlobalTempPerson  From Person;  Select \* From ##GlobalTempPerson -- Run this one line in a new tab  /\*\*\*\*\*\* Indexing a Temp Table \*\*\*\*\*/  DROP TABLE IF EXISTS #TempPerson;  Select ID, LastName, Firstname  Into #TempPerson  From Person;  CREATE NONCLUSTERED INDEX ix\_TempPerson ON #TempPerson (LastName)  Select \* From #TempPerson where LastName = 'Allscripts'  ???? where are they |

# Adding and Removing Rows

I think this whole section needs a disclaimer, especially when talking about delete, truncate, and drop

|  |  |
| --- | --- |
| Insert Into **Run the following queries**  There has been confusion around the behavior of SQL when you indicate the values you are asking and whether the select needs to be in the appropriate order when the columns are aliased/named properly in the query. | Insert Into Person  (LastName, FirstName, MiddleName, SexDE, SSN, IsInactiveFLAG, MungedLastName)  Values ('z{YourLastName}', 'Tom', '', 0, '','N', 'z{YourLastName}')  Select \*  from Person Where LastName = 'zTester'  --What happens in these situation  Insert Into NewPerson --\*Not a real table  (LastName, FirstName, MiddleName, SexDE, SSN, IsInactiveFLAG, MungedLastName)  Select MungedLastName,LastName, FirstName, MiddleName, SexDE, SSN, IsInactiveFLAG,  from Person Where LastName = 'zTester'  Insert Into NewPerson --\*No a real table  Select MungedLastName,LastName, FirstName, MiddleName, SexDE, SSN, IsInactiveFLAG,  from Person Where LastName = 'zTester' |
| Select Into **Run/Contrast/Compare the following queries**  Generally used for making full backup copies of tables  Note error if table already exists | SELECT \*  INTO zProvider\_bk\_20220101 -- Non-existant table  FROM Provider  SELECT column1, column2, column3, ...  INTO zProvider\_bk\_20220101 -- Non-existant table  FROM Provider |

|  |  |
| --- | --- |
| Delete **Run the following queries**  'z{YourLastName}' should exist from the previous INSERT exercise.  **\*\* BEST PRACTICE \*\***   * Use “Begin Transaction” and “Commit” as a safety check * Use a WHERE to avoid removing ALL ROWS * Transaction will lock a table, so be sure to commit or rollback.   Might be worth noting performance issues if deleting a large number of rows and maybe having to batch the deletion process. | Select \* from Person Where LastName = 'z{YourLastName}'  Begin Transaction  Delete Person Where LastName = 'z{YourLastName}' and ID = #####  Select \* from Person Where LastName = 'z{YourLastName}'  Rollback;  Select \* from Person Where LastName = 'z{YourLastName}'  Begin Transaction;  Delete Person Where LastName = 'z{YourLastName}' and ID = #####  Select \* from Person Where LastName = 'z{YourLastName}'  --Commit;  --Rollback; |
| Truncate | SELECT \* INTO zProvider\_bk\_20220101 FROM Provider  Select Count(\*) from zProvider\_bk\_20220101 -- 35 rows  Truncate Table zProvider\_bk\_20220101;  Rollback;  Did it work?  Select Count(\*) from zProvider\_bk\_20220101 -- 0 rows |
| Drop | Drop Table zProvider\_bk\_20220101 |

**Practicum COMPLETE**