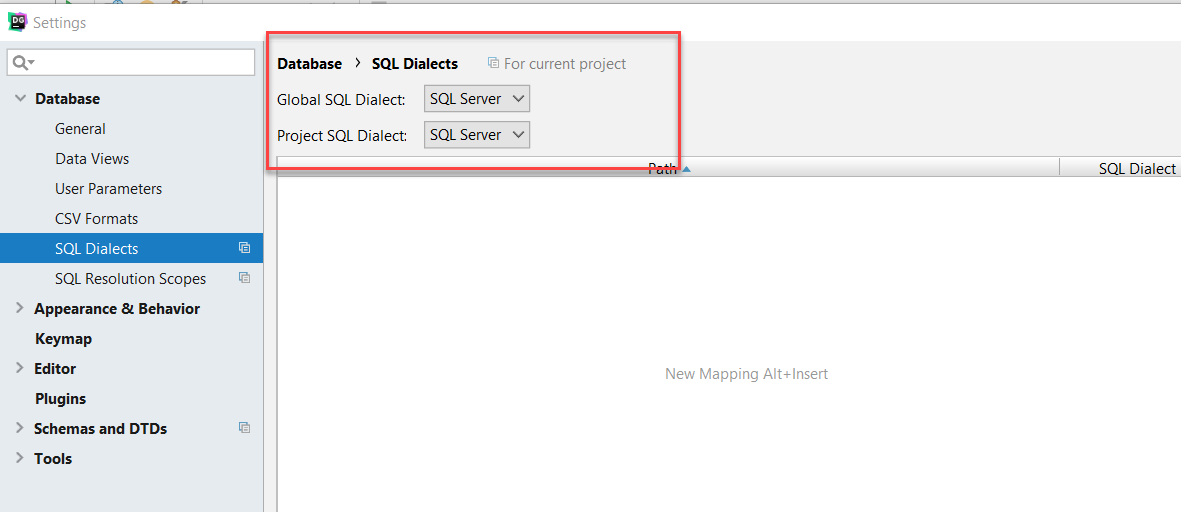
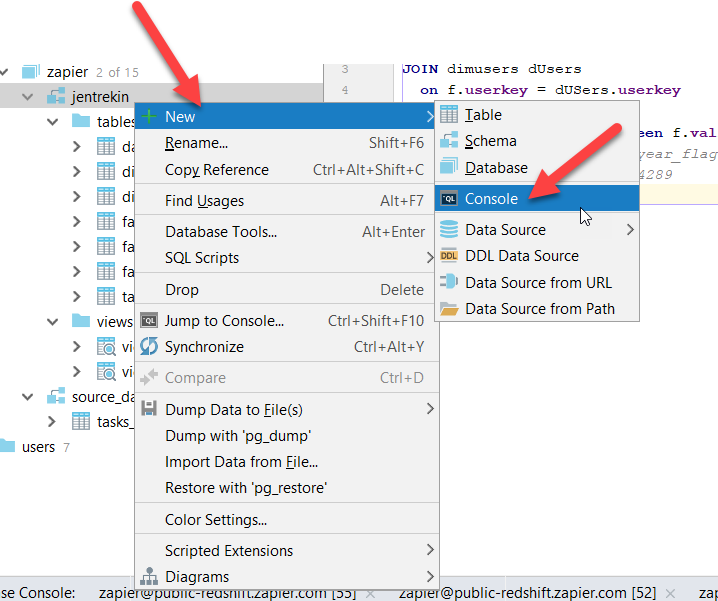
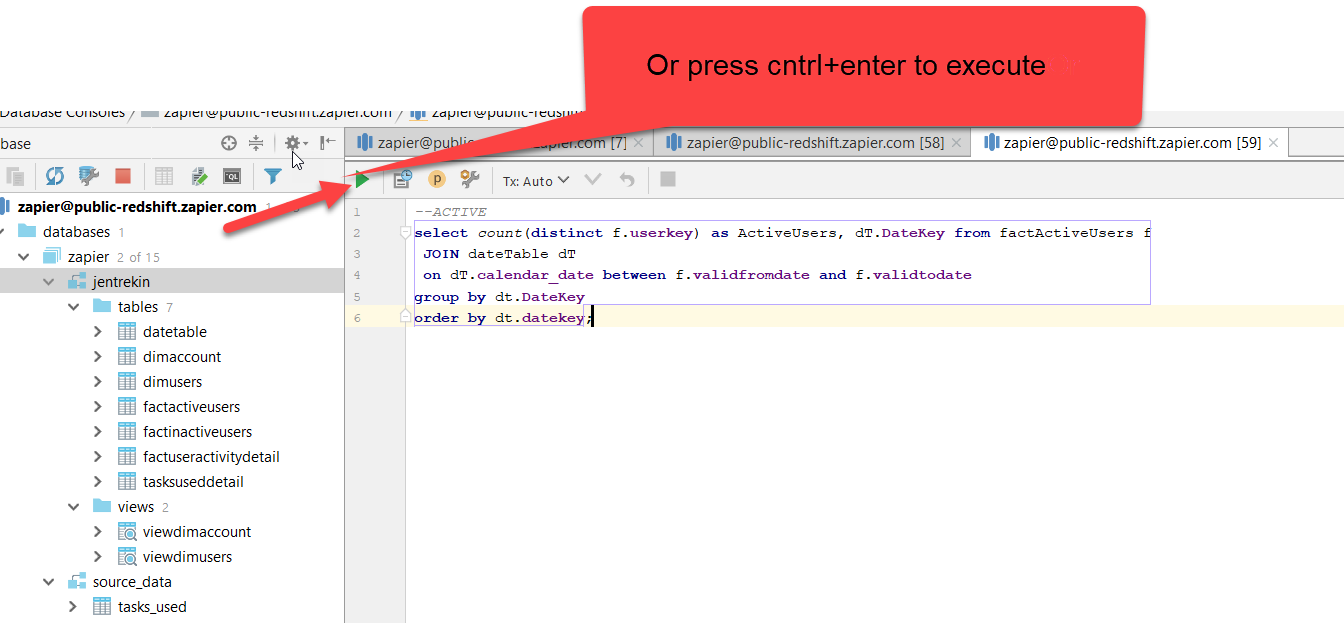
# Steps to deploy and execute

1. The following files are included in the GitHub submission:
   1. DateTableDDL.sql
   2. DateTableFill.sql -- populates the values for the date dimension
   3. dimAccountDDL.sql
   4. dimAccountFill.sql -- populates the values for the Account dimension
   5. dimUsersDDL.sql
   6. dimUsersFill.sql ---- populates the values for the Users dimension
   7. viewDimAccountDDL.sql
   8. viewDimUsersDDL.sql
   9. TasksUsedDetailDDL.sql
   10. TasksUsedDetailFill.sql -- populates the data from source\_data.tasks\_used
   11. factUserActivityDetailDDL.sql
   12. factUserActivityDetailFill.sql – populates the data from TasksUsedDetail
   13. factActiveUsersDDL.sql
   14. factActiveUsersFill.sql -- populates the data from factUserActivityDetail
   15. factInactiveUsersDDL.sql
   16. factInactiveUsersFill.sql -- populates the data from factUserActivityDetail
   17. ActiveCountMeasureByDay.sql –query that generates the active user counts by day
   18. ChurnCountMeasurebyDate.sql –query that generates list of churn users by day
2. The data model was created using DataGrip and using SQL Server SQL Dialect under File – Settings – Database – SQL Dialects – Global/Project SQL Dialect



1. Order of Deployment. In all of the below scripts, you’ll want to connect to the jentrekin schema, right click the schema name, and select New – Console. Then you’d want to paste the scripts into the new console window and hit either Cntrl+Enter OR the Play button.





* 1. Execute DateTableDDL.sql
  2. Execute DateTableFill.sql
  3. Execute dimAccountDDL.sql
  4. Execute dimUsersDDL.sql
  5. Execute dimAccountFill.sql
  6. Execute dimUsersFill.sql
  7. Execute viewDimAccountDDL.sql
  8. Execute viewDimUsersDDL.sql
  9. Execute TasksUsedDetailDDL.sql
  10. Execute TasksUsedFill.sql
      1. NOTE this script enables an incremental load, so the first time there’s no need for any modifications.
      2. Full load executes in approximately 2 minutes or less.
      3. However, if you in the future need to do a full reload, you would need to uncomment out the truncate statement and then comment out the highlighted lines of the code shown here:

*--TRUNCATE TABLE jentrekin.TasksUsedDetail***INSERT INTO** jentrekin.TasksUsedDetail ( **zap\_id**, **user\_id**, **userkey**, **account\_id**, **accountkey**, **date**, **datekey**, **min\_timestamp**, **tasks\_used**, **errors**)  
**SELECT** t.**zap\_id**, t.**"user\_id "**, *coalesce*(**d**.**userkey**, -1) **as** UserKey,  
 t.**account\_id**, *coalesce*(dAC.**accountkey**, -1) **as** AccountKey, t.**date**,  
 *cast*(*replace*(*cast*(t.**date as varchar**(10)), **'-'**, **''**) **as int**) **as** DateKey, t.**min\_timestamp**, t.**tasks\_used**, t.**errors  
from** source\_data.tasks\_used t **LEFT JOIN** jentrekin.dimusers **d on** t.**"user\_id "** = **d**.**user\_id  
left join** jentrekin.dimaccount dAC  
 **on** t.**account\_id** = dAC.**account\_id  
WHERE** t.**date** > (**SELECT** *isnull*(*MAX*(**date**), **'1900-01-01'**) **from** jentrekin.tasksuseddetail) *--set up incremental load*

* 1. Execute factUserActivityDDL.sql
  2. Execute factUserActivityDetail.sql
  3. Execute factActiveUsersDDL.sql
  4. Execute factInactiveUsersDDL.sql
  5. Execute factActiveUsersFill.sql
  6. Execute factInactiveUsersDDL.sql

1. Once the above deployment is complete, you should then be able to run the Churn and Active queries in steps 5 and 6 below.
2. Execute ChurnCountMeasureByDay.sql to get a count of distinct users who contributed to churn each day.
3. Execute ActiveCountMeasureByDay.sql to get a count of distinct users who are considered active each day.