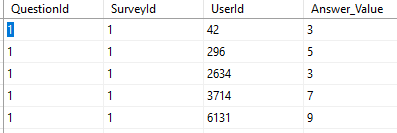
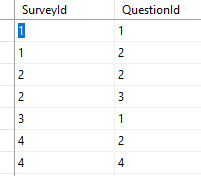
***Inputs in a SQL database :***

Answer table, which reports the answer of each user, the related question and survey :



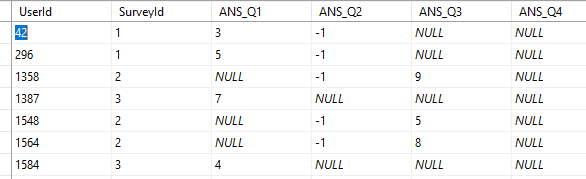
Survey structure, which details the questions included in each survey :



***Output to provide :***

AllSurveyData, which is a kind of rotated view of previous Answer table, but each existing question is illustrated by a column, in which there is :

* the answer number when answered
* "-1" when not answered, when question was inside survey to which user participated
* "NULL" question was not inside survey to which user participated



***Files organisation :***

* ***m\_trg\_refreshSurveyView.py*** which mimics a SQL trigger firing on INSERT, DELETE and UPDATE upon the table dbo.SurveyStructure. The refreshed view is then saved in a csv file (***vw\_AllSurveyData.csv***).
* ***m\_fn\_GetAllSurveyData.py*** which generates and returns a dynamic SQL query string for extracting the pivoted survey answer data.
* Connection to SQL database is handled by the generic connection DBconnector class (***m\_DBconnector.py***). Then SQLconnector is the specific class (**m\_SQLconnector.py**), which inherits from the previous, in order to handle database connection, on SQL server, with ODBC driver.

***Scope statement :***

* Access to the database server via an ODBC library.
* Recommended libraries are pyodbc and pandas.
* The Python application should not require the user to install packages before the run.
* Development in Python 3.
* Gracefully handle the connection to the database server.
* The last known surveys’ structures should be stored.
* Of course, extract the “always-fresh” pivoted survey data, in a CSV file, adequately named.