Female Speaker: Hello. My name is *Patricia Devassy[Phonetic]* and I am here today to discuss advanced transformer case studies. For the agenda, we will discuss cube build phases, improving build performance with some tips, using the Transformer log file for phase timing, transformer log keywords, autopartitioning, and then we will have an exam period where we will take the information we have learnt in this presentation and apply it to some specific scenarios. After that, we will have a bonus round question. For the first cube build phase, we will talk about the data read phase. Transformer reads the data source and creates a temporary work directory based on the structure of the Transformer model. Memory used climbs rapidly when categories are being generated during this phase. The more categories in the model, the more memory required. Reviewing this phase will tell you how long it took Transformer to read and process all the data sources and how many records were read. Keywords in the log file include initializing categories, open data source, read data source, and marking categories used. Associated problems with this phase can include database connectivity problems, slow read time, and insufficient disk space. The metadata update phase is the next phase in the transformer log file. The contents of the temporary work files are compared to the categories in the Transformer model to determine which categories will be put into the PowerCube. When the list of eligible categories is complete, the categories are inserted into the PowerCube. Keywords for this phase include sorting, update category and process work file, and metadata. Associated problems with this phase can be due to lack of memory, slow hard drive, and write cache settings. The next phase is the data update phase. The values in the temporary work files are inserted into the PowerCube. Each record inserted into the cube is a data point that consists of a category reference from each dimension in the model along with a measure of values for the intersection of those categories. Keywords for this phase include cube update and cube commit. Long commit times could be due to UDA sort memory settings, slow hard drives, inefficient partition settings or low system memory. Here are some tips on improving build performance. Choosing the fastest available processor speed should be considered as the addition of a second CPU can result in significant reduction in the data read phase. Optimally, there should be enough memory on the build computer to handle all running application request for memory and allow the operating system disk cache to grow as required. The most important choice *majoring[Phonetic]* hardware selection is memory followed closely by disk configuration. Excessive *paging[Phonetic]* will take place in situations where there is not enough physical memory available for Transformer, which will result in significant increase during the PowerCube build time. The more categories, the more memory required. For the read level, level 0 provides the fastest performance. The *type[Phonetic]* speeding configuration of the drive's subsystem can cause a significant increase in the time it takes to build a PowerCube. The recommended approach is for the first controller to have the operating system and applications, the second controller to contain the transformer data work directory and the third controller to contain the sort directory and PowerCube directory. A transformer log file is generated every time a model is processed. Using a spreadsheet program, this log file can be used to quickly understand the length of each of the three phases of the cube build. You would launch Excel and select file open for types of all files. You will see the dialogue box *below up here[Phonetic]* on the text import wizard. Keep the type set as delimited and select next. The dialogue box for step #2 of the import wizard appears. Make sure that tab and comma are both selected and then click Finish. The log file is then loaded into Excel and appears as follows. Select the entire E column and choose the data menu followed by the filter item and finally select the autofilter option. From the drop down list that appears in the E column, select nonblanks or a specific phase such as read data source or metadata. The spreadsheet now shows only the lines that contain timing information. Once the spreadsheet is in this format, select the *remainder of[Phonetic]* cells in the F column and look at the bottom of the Excel window to see the sum of the timing values. Transformer log keywords - as discussed the following displays the keywords that relate to each of the phases of the PowerCube build. Using the Excel spreadsheet to breakdown these phases will help you understand where the majority of time is being spent in each of your PowerCube builds. Now, I am going to move on partitioning. Partitioning presummarizes the data in the PowerCube and groups it into several subordinate partitions so retrieval will be significantly faster. As the number of partitions increase, the longer it will take to create the cube. The first partition pass will always pick the dimension with the highest number of categories. The next pass will pick the dimension with the second highest number of categories and so on. Please note that the first dimension in the model is always referred to as dimension 0 in the log file. So, when you are looking at your Transformer model in the UI, the first dimension on the left is considered dimension 0, not dimension 1 in the Transformer log file. So, on this slide we have an example of a partitioning pass. When looking at partitioning in a Transformer log file, there are three important sections you need to become familiar with. The first section I have highlighted in red shows the pass number, which is important as it tells you how many partitioning passes are being completed. In Transformer, you can specify the number of passes, the Transformer will only use up to the number of passes you specify. It won't necessarily use all of the passes. The next section I have highlighted in blue shows you that the dimension that Transformer is partitioning on. In this case, it has selected dimension 2 for the next pass of partitioning. We were looking at the model. You will see that it is the third dimension from the left. The next section is highlighted in green is very important as it will tell you if consolidation occurred. In this example, the start and end counts are the same, which means that no consolidation occurs and that this partitioning pass was unsuccessful. Now, in the next slide we are looking at the next pass. So, in this example, you see that Transformer selected dimension 3 for partitioning, which is dimension 4 when looking at the Transformer in the UI, the model in the UI, and the start and end counts are different, which means that consolidation occurred and the partitioning pass was successful. The next partitioning pass is pass #2. We have selected dimension 4 for this pass and the start and end counts are different. So, additional consolidation has occurred. Pass #3, we selected dimension #4 again in an attempt to provide additional consolidation and you can see with the start and end counts, additional consolidation has occurred. Onto pass #4, in this pass, Transformer selected dimension 1 and in this case, additional consolidation has been done, again, comparing the start and end counts. Pass #5 is the final pass where Transformer performs summary partitioning consolidation. The last row number in the log file is the number you want to input into the number of consolidated records. You can just round it up. In this example, we would round it up to eight million. So, if we take a look at each of the passes, the log file shows a total of five passes where Transformer attempts to provide consolidation. Comparing the passes shows that dimensions 2, 3, and 4 and 1 are dimensions where Transformer attempted partitioning. Taking a look at pass 0, you can see that the start and end counts were the same, which means that no partitioning occurs. This means that you can exclude this dimension for partitioning and drop the number of passes by one to save time during the cube build if your build time is a concern. Comparing the next two passes, you will notice that Transformer made two passes on dimension 0, which provided additional consolidation on the second pass. With partitioning it is important to take a look at the start and end counts. To determine if a cube is being partitioned well, compare pass 0 to pass 5 or the last pass listed in the log file. If the number is significantly lower, you can see that consolidation has been done. So, now we are going to talk about some different scenarios pertaining to the information that you have learned in this presentation. So, the first scenario we have is cube build takes much longer on one server versus another. So, on Unix the cube build takes 33 hours, on Windows the cube build takes a 100 hours, has the same model and the same database. The question is why does it take so much longer on Windows? When we take a look at the log files on Unix, we have a total time to create the cube of 33 hours and on Windows we have a total time to create the cube of 100 hours. So, the choices for this question are differences in hardware, database performance, multiprocessing isn't enabled, and category count is sufficiently higher. Correct answer is D. If we take a look at the log files, you can see that the Windows build has over 270000 more categories than the Unix build. Increasing the category build can significantly increase the amount of time it takes to build the PowerCube. Next scenario - the cube build fails at the same location every time. So, it’s a Unix build failing during the data read phase, remember that important point, data read phase, all previous cube builds work, it's the same model and the same database. So, the question is why is the cube build now failing? We take a look at the Transformer log files, there is two error messages. TR0112, there isn't enough memory available and REPOSE NOMEM Insufficient Memory on the Machine for operation. The question is why is the cub build now failing? So, A) the computer needs to take a break at the same point every time, it's worn out and tired, B) a limitation has been reached, C) multiprocessing wasn’t enabled, and D) database connectivity problems. The correct answer is B. If you take a look at the transformer log file, you can see that there are over two million categories listed. The supported limit for Transformer is two million categories. After redesigning the model, we no longer exceeded the limit and the Powercube built to completion. The next scenario - it’s a long cube build with a limited data set. So, the Windows cube build was completing successfully, but the PowerCube was taking too long for the number of input records and categories. So, why is the cube build taking almost eight hours to complete? The log file confirmed we had under a three million records and 550000 categories, but it was taking seven-and-a-half hours to complete the cube build. This is a fairly low record count and category count. So, the fact that it is taking seven-and-a-half to eight hours to complete the cube build is really long. So, the options are low system memory, inefficient preference settings, autopartitioning isn't being utilized, and the database read is slow. The correct answer was C. We took a look at the Transformer log file. There was an error message in the log file that says, "This model contains one or more cubes that use a dimension view in which the primary drill down is cloaked." autopartitioning Is not possible when a primary drill down is cloaked. So, we changed the primary drill down and rebuilt the cube and the cube now builds in 40 minutes. So, it’s really important to scan your

Transformer log files and look for error messages or warning messages. So, the next question is which pass is most successful at partitioning? So, if you take a look at this slide, you have A, B, and C to choose from. It's which pass is most successful at partitioning? The correct answer is C. If you look at the start count and the end count, the most consolidation has occurred during this pass. Now, can you tell me which pass is the least successful at partitioning? The pass that is least successful at partitioning is B. Comparing the start and end count shows that absolutely no consolidation occurred. So, this is the bonus round question. Based on the partition results from the last slide, what can you do to reduce the PowerCube build time without degrading end user performance? So, again, I am showing the start and end counts. We know that B or the second pass we show here actually has absolutely no consolidation. So, in this scenario, what would you do to reduce PowerCube build time? The correct answer will be you would exclude this dimension which happens to be dimension 0 from partitioning and this would reduce your PowerCube build time. So, that's the end of the presentation. If you have any questions, please send an e-mail to *patricia.devassy@ca.ibm.com[Phonetic]*. Thank you for your time.