

OLAP and Data Warehousing

Lab Exercises, SS 2015

Part I OLAP

Purpose:

The purpose of this practical guide to data warehousing is to

- learn how online analytical processing (OLAP) methods and tools can be used to perform multidimensional analysis of data,
- get hands-on experience and skills in using MS SQL Server Integration Services (SSIS) and Analysis Services (SSAS) tools for building Extract-Transform-Load (ETL) processes and for building multidimensional data cubes.

The task consists in building a simple OLAP tool for analysis of student notes. The tool will be used to investigate interesting relationships such as e.g., how student notes depend on such variables as teacher seniority ('Are older teachers more strict than younger teachers?'), time, faculty, type of course, and how this is changing over time, etc.

Input data:

Data sets:

```
notes.csv
students.csv
teachers.csv
teacher_title.csv
course_group.csv
```

In the tutorial part of the lab we will use the fhhnotes.mdb database.

Tasks – overview:

1. Using Integrated Services Project, design ETL process to load source data (the *.csv files) into a SQL database.
Clean / modify / integrate data using SSIS data flow transformations and/or SQL tasks.
2. Using Analysis Service project, design / deploy multidimensional cube with:
 - Note as the fact variable.
 - All other columns available in data to be used as analysis dimensions (dimension attributes). The cube should have the following dimensions (dimension attributes):
 - o teacher related, e.g., teacher's gender, title, affiliation,
 - o course related,
 - o student related,
 - o time / semester related,

- teacher or student workload (calculated per semester).
- Additional requirements:
 - Please ensure that the values in the note column are between 1 and 5 (remove illegal values).
 - Empty values in dimension columns should be substituted with some descriptive information (e.g., 'unknown').
- 3. Browse the cube in order to observe interesting relationships related to how notes depend on such classification variables as, e.g.: (consider various note statistics such as the average, number of values, range, etc.):
 - teacher senioriy (teach_duration),
 - course type,
 - job title of teacher,
 - gender of teacher,
 - probation,
 - individual teacher number,
 - time,
 - workload of teacher.

Produce a report summarizing your most interesting findings.

Hints

Detailed how-to procedures and hints are given in the Script **DATA MINING AND DATA WAREHOUSING – PRACTICAL GUIDE**, part II.

Also refer to the two technical sections below:

DETAILED DESCRIPTION OF SUBTASKS

HINTS AND HOW-TO PROCEDURES

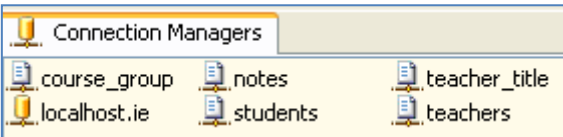
Detailed description of subtasks:

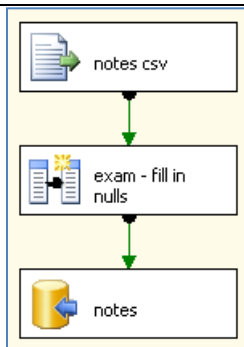
Task	Description																																						
1	<p>Import source data</p> <p>Using MS VS Integration Services project, load the source files notes.csv, students.csv, teachers.csv, course_group.csv, teacher_title.csv into MS SQL database. Meaning and type of each column is explained below.</p> <table> <tr> <th colspan="2">Table notes</th></tr> <tr> <td>Semester</td><td>Semester number Integer in range 1..10</td></tr> <tr> <td>Year</td><td>Calendar year Integer in range 1980..</td></tr> <tr> <td>Course</td><td>Id of course Text, up to 10 characters in length</td></tr> <tr> <td>teacher_id</td><td>Id of a teacher who granted a note to a student Integer in range 0..99999</td></tr> <tr> <td>Note</td><td>A note obtained by a student in a course Real value, legal values: 2, 3.0, 3.5, 4.0, 4.5, 5.0, 5.5</td></tr> <tr> <td>Exam</td><td>Value 'E' identifies a note obtained in an exam, empty value identify notes based on course-work performance of a student.</td></tr> <tr> <td>student_id</td><td>Id of a student who received the note Integer in range 0..999999</td></tr> </table> <table> <tr> <th colspan="2">Table students</th></tr> <tr> <td>Spec</td><td>Specialization of the student (AIR – Automatics & Robotics; EIT – Electronics & Telecommunication; INF – Computer Engineering) Text, 3 characters in length</td></tr> <tr> <td>sub_spec</td><td>Sub-specialization within specialization groups Text, 3 characters in length</td></tr> <tr> <td>sub_spec2</td><td>Some (rare) sub-specializations are further split into groups defined by this value. For most sub-specializations this column is empty. Text, 3 characters in length</td></tr> <tr> <td>Gender</td><td>Gender of a student 'K' – female, 'M' – male</td></tr> <tr> <td>student_id</td><td>Id of a student, PK of this table Integer in range 0..999999</td></tr> </table> <table> <tr> <th colspan="2">Table teachers</th></tr> <tr> <td>teacher_id</td><td>Id of a teacher, PK of this table Integer in range 0..99999</td></tr> <tr> <td>Gender</td><td>Gender of a teacher 1 – male, 2 – female</td></tr> <tr> <td>Faculty</td><td>Id of the Faculty the teacher is affiliated with (value of 4 denotes Faculty of Electronic Engineering) Integer in range 1..12</td></tr> <tr> <td>Institute</td><td>Id of the Institute the teacher is affiliated with Text, up to 5 characters in length</td></tr> </table>	Table notes		Semester	Semester number Integer in range 1..10	Year	Calendar year Integer in range 1980..	Course	Id of course Text, up to 10 characters in length	teacher_id	Id of a teacher who granted a note to a student Integer in range 0..99999	Note	A note obtained by a student in a course Real value, legal values: 2, 3.0, 3.5, 4.0, 4.5, 5.0, 5.5	Exam	Value 'E' identifies a note obtained in an exam, empty value identify notes based on course-work performance of a student.	student_id	Id of a student who received the note Integer in range 0..999999	Table students		Spec	Specialization of the student (AIR – Automatics & Robotics; EIT – Electronics & Telecommunication; INF – Computer Engineering) Text, 3 characters in length	sub_spec	Sub-specialization within specialization groups Text, 3 characters in length	sub_spec2	Some (rare) sub-specializations are further split into groups defined by this value. For most sub-specializations this column is empty. Text, 3 characters in length	Gender	Gender of a student 'K' – female, 'M' – male	student_id	Id of a student, PK of this table Integer in range 0..999999	Table teachers		teacher_id	Id of a teacher, PK of this table Integer in range 0..99999	Gender	Gender of a teacher 1 – male, 2 – female	Faculty	Id of the Faculty the teacher is affiliated with (value of 4 denotes Faculty of Electronic Engineering) Integer in range 1..12	Institute	Id of the Institute the teacher is affiliated with Text, up to 5 characters in length
Table notes																																							
Semester	Semester number Integer in range 1..10																																						
Year	Calendar year Integer in range 1980..																																						
Course	Id of course Text, up to 10 characters in length																																						
teacher_id	Id of a teacher who granted a note to a student Integer in range 0..99999																																						
Note	A note obtained by a student in a course Real value, legal values: 2, 3.0, 3.5, 4.0, 4.5, 5.0, 5.5																																						
Exam	Value 'E' identifies a note obtained in an exam, empty value identify notes based on course-work performance of a student.																																						
student_id	Id of a student who received the note Integer in range 0..999999																																						
Table students																																							
Spec	Specialization of the student (AIR – Automatics & Robotics; EIT – Electronics & Telecommunication; INF – Computer Engineering) Text, 3 characters in length																																						
sub_spec	Sub-specialization within specialization groups Text, 3 characters in length																																						
sub_spec2	Some (rare) sub-specializations are further split into groups defined by this value. For most sub-specializations this column is empty. Text, 3 characters in length																																						
Gender	Gender of a student 'K' – female, 'M' – male																																						
student_id	Id of a student, PK of this table Integer in range 0..999999																																						
Table teachers																																							
teacher_id	Id of a teacher, PK of this table Integer in range 0..99999																																						
Gender	Gender of a teacher 1 – male, 2 – female																																						
Faculty	Id of the Faculty the teacher is affiliated with (value of 4 denotes Faculty of Electronic Engineering) Integer in range 1..12																																						
Institute	Id of the Institute the teacher is affiliated with Text, up to 5 characters in length																																						

	title_id	Id of teacher's title, used to relate to the teacher_title table Integer in range 0..40
	Table course_group	
	Course	Id of a course, PK of this table Text, up to 10 characters in length
	course_group	1 – science and technical courses 2 – sports 3 – foreign languages 4 – social science and management courses
	Table teacher_title	
	title_id	Id of teacher's title, PK of this table Integer in range 0..50
	title_long	Teacher's full title Text, up to 30 characters in length
	Title	Teacher's short title Text, up to 10 characters in length
2	<p>Fix data value inconsistencies / unclear coding conventions</p> <p>Some columns contain wrong values (e.g., note = 0 or 1), these values should be removed. Some columns also contain missing values, these should be filled in with some meaningful data (e.g., empty exam column can be filled in with text 'CW' (course work)). Empty values interpreted as information not available should be changed into text 'unknown' or '?', etc. Gender in teachers and student tables should be coded consistently using symbols clear to end user.</p>	
3	<p>Fix consistency of relationships between tables</p> <p>Consistency of foreign key – primary key (PK) relationships between tables should be verified, and, if needed, corrected. For instance, teacher_id in notes table is FK to teachers table. However, source data is not consistent and this relationship is broken as some values of FK (teacher_id) in notes table are missing in teachers table. The way to correct this is to add the missing teacher_id entities to the teachers table. All remaining attributes for these teachers should be coded as unknown (gender, faculty, institute, title).</p> <p>The same procedure should be carried out for FK – PK relationships between other tables (notes – course_group, etc.).</p>	
4	<p>Derive new dimension attributes</p> <p>Several interesting attributes are not directly available in source data, e.g., semester type (winter semesters vs summer semesters). Such new columns should be derived, which later allows to add new interesting dimension attributes to the cube.</p> <p>Proposed new attributes:</p> <ul style="list-style-type: none"> Semester type (winter/summer semesters based on whether semester number is odd/even) Academic year of studies (semesters 1,2 – 1st year, semesters 3,4 – 2nd year, etc.) Course type – based on the last character of the course symbol (W=Lecture, L=lab, P=Project, S=Seminar, C=Exercise). 	

5	<p>Derive workload table (workload dimension)</p> <p>A new dimension that describes per-semester workload of teachers (and/or workload of students) should be derived. Workload can be defined as the number of notes a teacher grants to the students in a given semester, or, alternatively, as the number of different courses a teacher gives in a semester. The teacher workload table should be created with the columns teacher_id and semester used as the composite PK.</p>
6	<p>Restructure the fact table notes</p> <p>The fact table should include only the fact variable (note) and FKs to dimension tables. This can be achieved by moving non-FK columns from the fact table to a separate table (denoted as notes_desc – see Hint 5) and placing the FK to this table in the fact table (denoted as notes_desc_id).</p>
7	<p>Create a multidimensional cube</p> <p>Using MS VS Analysis Services project, a cube should be designed with the notes fact table and dimension tables pertaining to students, teachers, courses, teacher workload, and other notes description (such as those created in task 2).</p>
8	<p>Edit measures in the cube</p> <p>The cube should allow for analysis of the following statistics: the number of notes and the average note. Thus, the measures in the cube should include sum of notes and count of notes. A calculated member average note should be then defined for the cube, based on ratio of these two measures.</p>
9	<p>Edit dimensions</p> <p>Edit all the dimensions (teacher, student, course, notes descriptions and workload dimension) by adding relevant attributes. Proposed structure of dimensions is shown in Hint 5. Related dimension attributes should be grouped into drill-down hierarchies, e.g., Specialization hierarchy, which groups Spec→Sub Spec→Sub Spec2 attributes. Notice: good practice should be observed to hide in the cube structure dimension attributes used as levels of hierarchies.</p>
10	<p>Build and deploy the cube</p> <p>Once designed, the cube should be deployed on the instance Analysis Services. The cube is ready to be browsed with an OLAP viewer or queried using MDX language.</p>
11	<p>Analyze the cube and demonstrate some interesting relationships / trends pertaining to notes at our Faculty.</p>

Hints and how-to procedures:

Hint	Problem / solution
1	<p>How to import source csv files to MS SQL database?</p> <ol style="list-style-type: none"> 1. Using SQL Server Data Tools (SSDT) create new Integration Services project. 2. Create new Flat File Connection Manager (right click in Connection Manager pane). Edit this Connection Manager to provide location of the source file as well as formatting details (such as column delimiters, etc.). Care should be taken to specify the proper type of each column – compliant with the information detailed in Task 1. 3. Create new OLE DB Connection Manager. This Connection Manager is used to specify name of the database used to store imported tables. 4. Create new Data Flow Task (drag and drop Data Flow icon from toolbox onto the Control Flow pane). 5. Put Flat File Source into this Data Flow Task and specify the appropriate Flat File Connection Manager. 6. Put OLE DB Destination into this Data Flow Task and connect Flat File Source to this OLE DB Destination. Edit the OLE DB Destination to specify right OLE DB Connection Manager and request New table to store imported data. Notice that this will create the destination DB table, with the generated SQL CREATE TABLE statement shown after pressing the New (table or view) button. It should be verified that the name of the destination table and types of columns are correct (the name of the destination table is taken after the name of the OLE DB Destination tool). 7. To import source csv file into the DB table, run the Data Flow Task (right click on its icon). <p>Repeat steps 2,4-6 of this procedure for the remaining source files. As a result, five Flat File Source Connection Managers, and one OLE DB Connection Manager will be created, as shown below. The Data Flow Tasks created can be grouped using Sequence Container (as shown in the Script, Figure 3).</p>  <p>Notice Care should be taken how null values in source tables are loaded using Flat File (FF) Source tool. When the FF Source option “Retain null value” is checked, nulls (coded as ,, in csv files) will be loaded as NULL into the database. This makes it possible to use constructs like WHERE teacher_is IS NULL or expressions like ISNULL(teacher_id), etc. (see Hint 2).</p>
2	<p>How to clean imported data / modify wrong values / fill in null values?</p> <p><u>Using Data Transformation tools in Data Flow Task</u> New columns can be derived or existing columns can be conditionally modified using Derived Column in the data transformation flow, as shown below. The expression shown in this example is used to overwrite null values in the exam column with an explicit symbol of course-work notes (“CW”, as opposed to exam notes, coded explicitly by “E”).</p>



Derived Column Name	Derived Column	Expression
exam	Replace 'exam'	ISNULL(exam) ? "CW" : exam

Rows can be conditionally deleted from a table using Conditional Split transformation.

Using explicit SQL code

Data cleaning transformations can be alternatively written as SQL scripts placed in Execute SQL Task (elements in Control Flow). For instance, below is a simple script used to recode teacher's gender:

```
UPDATE teachers
SET gender =
CASE
    WHEN gender = '1' THEN 'M'
    WHEN gender = '2' THEN 'F'
    ELSE '?'
END
```

3

How to fix foreign key – primary key relationships between tables

The problem will be explained on the example of the dimension teachers (FK – PK relationship between notes fact table and teachers table).

Verification of consistency of FK – PK relationship

It should be checked if any of values of the FK (teacher_id) in the fact table are missing in the dimension table, if so (as in our source data) – the relationship is broken.

The idea of SQL code used for verification of the relationship is given below.

```
SELECT n.teacher_id
FROM notes as n
LEFT JOIN teachers AS t ON n.teacher_id = t.teacher_id
WHERE (t.teacher_id IS NULL)
GROUP BY n.teacher_id
```

Correction of inconsistent relationship

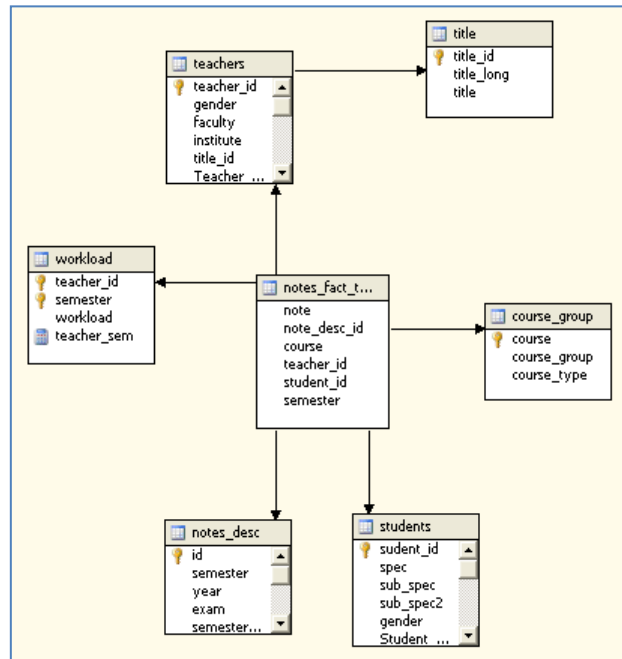
Teacher ids missing from the teachers table should be added to teachers dimension, forming entities with the remaining columns (teacher gender, etc.) filled with values coding 'information not available'.

The idea of an SQL script to correct broken relationships is given below.

```
INSERT INTO teachers
SELECT n.teacher_id, '?' AS gender, etc...
FROM notes as n
LEFT JOIN teachers AS t ON n.teacher_id = t.teacher_id
WHERE (t.teacher_id IS NULL)
GROUP BY n.teacher_id
```

4	<p>How to create dimensions based on non-FK columns of the fact table</p> <p><u>Method 1</u></p> <p>The idea is to create a new dimension table with non-FK columns of the fact table (such as e.g., semester, etc.; this table is named notes_desc in Hint 5). FK to this table will be placed in the fact table in place of the columns in this new dimension (these columns will be removed from the original notes table).</p> <p>The first step is to create a temporary (aggregate) notes table with the id used as the PK in the new dimension table and FK in the new fact table. The table will be then split into the notes fact table and the new dimension table.</p> <p>The idea of the first step is given below.</p> <pre> SELECT IDENTITY (bigint, 1, 1) AS id, all remaining columns of the fact table... INTO notes_tmp FROM notes </pre> <p>Then the temporary table should be split into the new dimension table:</p> <pre> SELECT id , semester, etc. INTO notes_desc FROM notes_tmp </pre> <p>and into the fact table (notice that only the fact variable and FKs to dimension tables are being selected):</p> <pre> SELECT note, id AS note_desc_id, remaining FKs to other dimension tables INTO notes_fact_table FROM notes_tmp </pre> <p>The multidimensional model of data formed in this way is shown in Hint 5.</p> <p><u>Method 2</u></p> <p>The idea is to create separate dimension tables for each of the non-FK column of the fact table while using this column in the fact table as the FK to the newly created dimension. The idea will be explained using the semester column as an example.</p> <p>The code to create new semester dimension:</p> <pre> SELECT DISTINCT semester, other semester-related columns... FROM notes </pre> <p>The new dimension table can be created either physically in the database (ETL script), or as an named query in Analysis Services data source view.</p>
5	How to create a multidimensional cube?

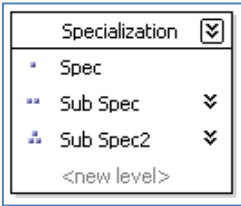
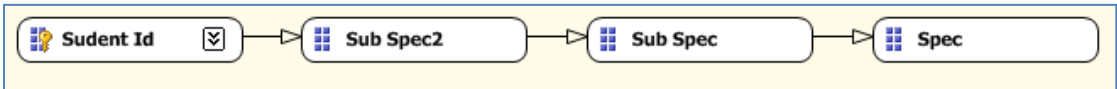
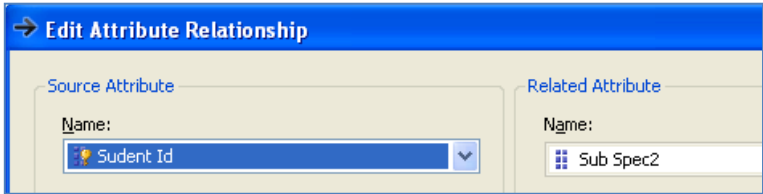
1. Using SQL Server Data Tools (SSDT) create new Analysis Services project.
2. Create a new data source (right click on Data Sources in Solution Explorer), pointing at the database created in the ETL stage. Consider selecting service account for impersonation information.
3. Create a new data source view (right click on Data Source Views), using the data source created in the previous step. Create logical FK-PK relationships between the fact table and the dimension tables (right click on a table, select New Relationship, specify fact table as the source (foreign key) table and a dimension table as the destination table). An exemplary data source view is shown below.



4. Create a new cube (right click on Cubes in Solution Explorer). Use the table notes as the measure group (i.e., fact) table, and remaining tables as dimension tables.

The cube can now be built and deployed in Analysis Services instance. However, to make the cube fully functional, measures stored in the cube as well attributes and properties of dimensions should be edited (see Hints 6, 7).

6	<p>How to add/edit measures stored in the cube?</p> <p>To add a new measure, right click on the Measures pane in the cube designer. When defining the measure, specify proper usage for the measure (such as sum or the number of notes, which defines how the fact variable should be aggregated when stored in the cube). See also the AggregateFunction property of the measure.</p> <p>Notice that only distributive measures can be defined as cube measures (such as sum or count of values of the fact variable). To provide a definition of non-distributive algebraic measure (such as the average), a calculated member should be defined for the cube (Hint 7).</p>
7	<p>How to add a calculated member to the cube?</p> <p>A calculated member allows to add an algebraic formula based on measures stored in the cube (e.g., average note based on sum and count of notes). To add a calculated member,</p>

	<p>select Calculations tab in the cube designer. Then in the right panel define name of the calculated member and an expression used to calculate it. E.g., the following expression calculates the mean note based on the two cube measures:</p> <pre>[Measures].[Note Sum]/[Measures].[Note Count]</pre> <p>Notice that while defining the formula, names of cube measures can be dragged from the Metadata pane of Calculations editor and dropped onto the Expression editor.</p>
8	<p>How to edit dimensions?</p> <p>In dimensions editor, the following properties of dimensions are specified:</p> <ul style="list-style-type: none"> • List of attributes of the dimension, • Drill-down hierarchies in the dimension, • Properties of attributes related to visibility, sort order, discretization method, etc. <ol style="list-style-type: none"> 1. To enter dimension designer, in cube designer expand the given dimension in question (in Dimensions pane), then select command Edit dimension. 2. To design attributes of a dimension, drag and drop a column name to be used as an attribute from the Data Source View pane of dimension designer onto the Attributes pane. 3. To create drill-down hierarchies, drag attributes to be grouped onto the Hierarchies pane, as in the example shown below (Specialization hierarchy of Students dimension).  <ol style="list-style-type: none"> 4. For attributes included in a hierarchy, define attribute relationships, as shown in the example below (use Attribute Relationships tab of dimension designer).   <ol style="list-style-type: none"> 5. To hide some attributes of a dimension, i.e., make them invisible for the user in OLAP browser, set the AttributeHierarchyVisible property of the attribute to False. For clarity of the multidimensional model presented to the user, attributes included in levels of drill down hierarchies are generally not shown as dimension attributes. 6. If members of an attribute are sorted improperly in the OLAP browser, the sort order can be changed using OrderBy attribute property. 7. For attributes with a large number of members (different values), such as the workload dimension, consider setting the DiscretizationMethod property (and related

	<p>DiscretizationBucketCount property). This leads to automatic grouping of the large number of values into buckets and improves clarity of presentation, as shown below.</p> <table><tr><th>Workload ▼</th><th>Note Avg</th><th>Note Count</th></tr><tr><td>1 - 18</td><td>4.17</td><td>12311</td></tr><tr><td>19 - 76</td><td>4.15</td><td>29070</td></tr><tr><td>77 - 129</td><td>4.14</td><td>11071</td></tr><tr><td>132 - 344</td><td>3.89</td><td>12742</td></tr><tr><td>Grand Total</td><td>4.10</td><td>65194</td></tr></table>	Workload ▼	Note Avg	Note Count	1 - 18	4.17	12311	19 - 76	4.15	29070	77 - 129	4.14	11071	132 - 344	3.89	12742	Grand Total	4.10	65194
Workload ▼	Note Avg	Note Count																	
1 - 18	4.17	12311																	
19 - 76	4.15	29070																	
77 - 129	4.14	11071																	
132 - 344	3.89	12742																	
Grand Total	4.10	65194																	
9	<p>How to build / deploy / browse the cube</p> <p>Once designed, the cube can be built and installed on the instance of Analysis Services (server of multidimensional databases). To do this, use Build → Deploy menu command in BI Development Studio.</p> <p>To perform data analysis using the cube, a multidimensional OLAP browser built into the development environment can be used. To launch the browser, open MS SQL Management Studio, connect to Analysis Services, right click the cube name and select the Browse command.</p>																		
10	<p>Security issue: if deployment of the cube fails due to insufficient rights of Analysis Services attempting to access the ETL database, then you need to grant the LOCALHOST\MSSQLServerMSASUser the db_datareader role for the ETL database.</p> <p>To do this, follow this procedure:</p> <ol style="list-style-type: none">1. In Management Studio connect to the SQL Server engine2. Select Security – Logins – New Login3. Login Name – Search4. Object Types – select Groups5. Advanced – Find Now6. Select SQLServerMSASUser login7. In Login Properties for this login, select User Mapping8. Select the ETL database (check the map box) and select the db_datareader role																		