DWH281 Project

Glory[600646] | Tokelo[601009] | Moloko[601352] | Yandile [601553]

Requirements gathering

- End users of the data warehouse that is to be designed include the staff members, lecturers as well as the Dean.
- The data warehouse shall be used to capture the progress of students' efforts in final class tests per semester at Belgium Campus iTversity.
- The Dean of this private institution shall gather information of the average performance of students in class tests within a particular task by initiating interviews with lecturers regarding test performance, lecture attendance and assessment completion.
- Questions asked in the data warehouse to the end users should capture information about the number of students that have achieved well in exams, compared to assignments and weekly tests, percentage of students that qualified for mid-term and final exams, comparison of the average grades of different groups of exams within the same module, correlation between accomplishment of success and type of high school the student graduated from, e.g., IEB or Cambridge, questions with more correct answers and difficulty of questions.as well as the memorandum.
- The memo is created for the teaching staff to provide to students for finding mistakes made in tests, exams and or assignments to learn lessons and get them captured for future mid-term and final exams. This can improve the average overall student performance.
- Recommended textbooks are provided for students to prepare for tests, assignments, and exams, especially mid-terms and final exams.
- Daily average work is set to qualify students for mid-term and final exams.
- Daily average work composes of the average mark of the final summative test(Testing of all work covered in the whole module), total number of weekly tests, total number of assignments and/or projects covered in a module, as well as lecture attendance
- Students do not qualify for mid-term exams if the students lecture attendance is below 85% OR the student achieves less than 50% for their daily average work.
- Daily Average work counts 30% of final mark, whereas mid-term and final exams count 70% of the final marks.

Definition of Users' Requests

Establishing user requirements plays a crucial role in the formulation of a data warehouse design. This entails comprehending the information requirements of end users and the nature of queries they intend to execute within the data warehouse. In the context of this initiative, the primary end users are the teaching staff and other interested members of the staff at

Belgium Campus. By delving into and specifying these user requests, a tailored data warehouse can be crafted to efficiently meet the needs of end users. This, in turn, empowers them to derive insights from data, facilitating informed decision-making and enhancing the teaching and learning experience at Belgium Campus.

These requests may include performance analysis, attendance and assessment impact, student performance trends, course performance metrics and exam question analysis.

Here's how one can group it:

Performance Analysis	Users might want to analyse students' performance across various dimensions, such as courses, exam types, and time.	"Show me the average scores of students in a specific course or across all courses." "How many students passed a particular course or all courses in a semester?" "Which courses have the highest or lowest pass rates?"
Attendance and Assessment Impact	Users might be interested in understanding the relationship between lecture attendance, weekly assessment scores, and exam performance.	"How does lecture attendance correlate with students' performance in exams and assessments?" "How do weekly assessment scores compare to mid-term and final exam scores?"
Student Performance Trends:	Users might want to identify trends in student performance across different courses or semesters.	"Are there any noticeable trends in student performance across different courses or semesters?" "Which students consistently perform well or struggle in their courses?"
Course Performance Metrics:	Users might want to compare course performance metrics between different academic years or semesters.	Which courses have the highest or lowest average weekly assessment scores?" "Which courses have the highest or lowest lecture attendance rates?"

Exam Question Analysis:	Users might want to analyse the exam questions to improve the quality of assessments.	"What is the correctness of the answers to the questions on the exams?" "Which questions have the majority of correct answers, and which are most difficult to answer?"

Our Data Warehouse consists of six dimension tables and three fact tables designed to capture various aspects of student performance, exam details, and course assessments.

<u>Defining the Fact Tables</u>

CoursePerformance_Fact: This fact table captures student performance metrics
StudentID(Foreign Key) CourseID(Foreign Key) Semester ExamScores Attendance WeeklyAssessmentScores TotalScores, Pass_Fail_Status Date ExamID(Foreign Key)

ExamResults_Fact: This fact table stores exam-related information.
StudentID(Foreign Key) ExamID(Foreign Key) ExamType ExamScores CorrectAnswers DifficultyLevel AverageScores Pass_Fail_Status QuestionID(Foreign Key)

AssessmentResult_Fact: This fact table contains course assessment metrics
StudentID(Foreign Key)

AssessmentID(Foreign Key)
AssessmentType
AssessmentScores
AverageScores
Pass_Fail_Status

Defining The Dimensional Tables

Student_Dim
StudentID (Primary Key)
Name
HighSchoolTypeID(Foreign Key)

Course_Dim
CourseID (Primary Key)
Name
Credits
Lecture_ID(Foreign Key)

Exam_Dim
ExamID (Primary Key)
ExamType
Date

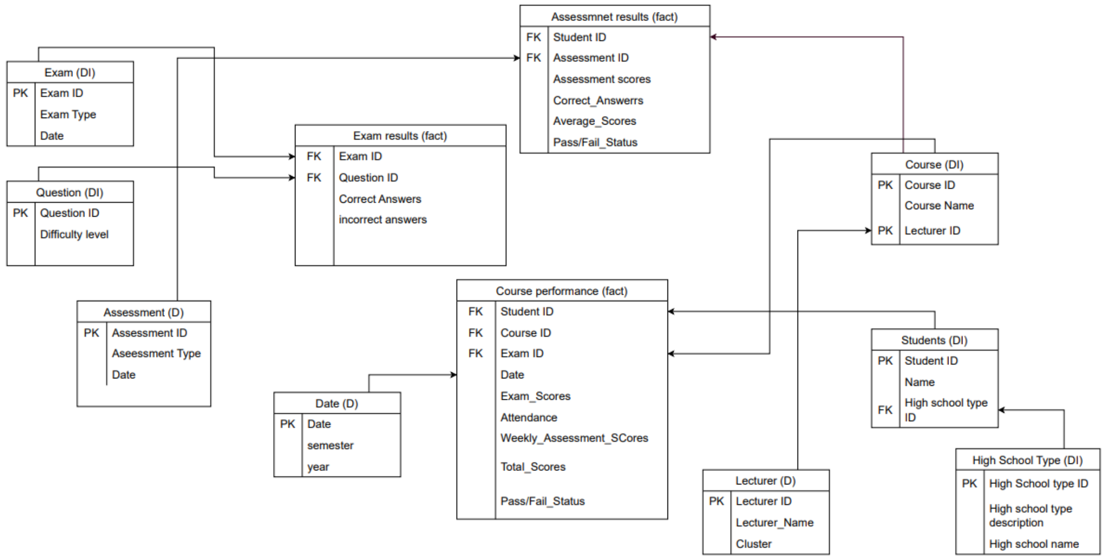
Assessment_Dim
AssessmentID (Primary Key)
AssessmentType
Date

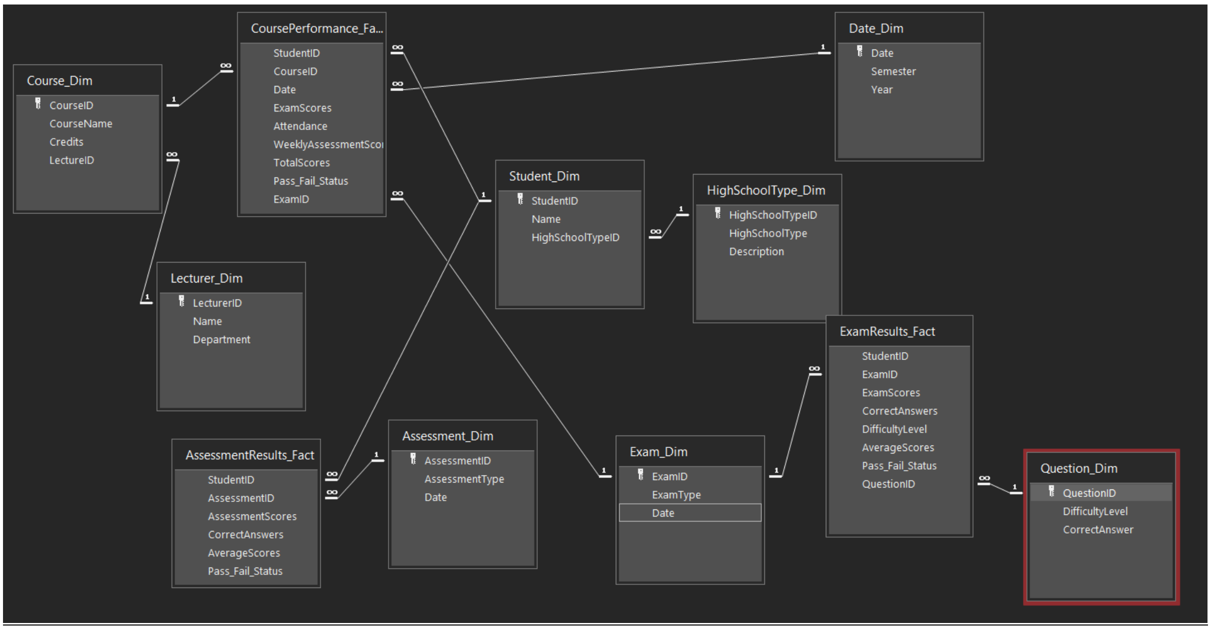
Date_Dim
Date (Primary Key)
Semester
Year

Lecturer_Dim
LecturerID (Primary Key)
LecturerName
Department

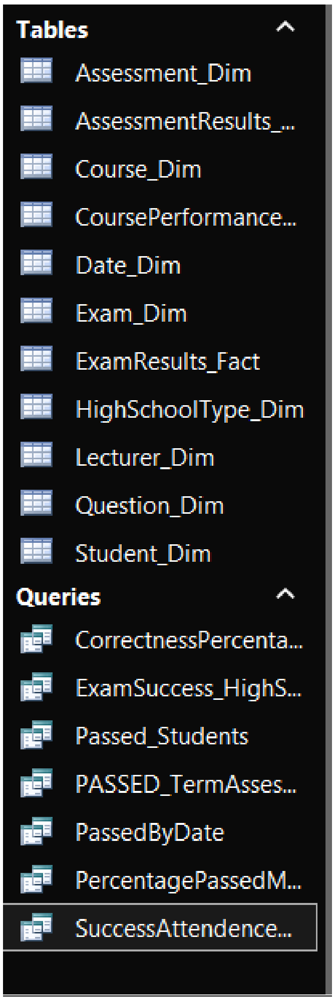
Question_Dim
QuestionID (Primary Key)
DifficultyLevel
CorrectAnswer

Snowflake Schema





Tables and Queries



Tables

- Assessment_Dim
- AssessmentResults_...
- Course_Dim
- CoursePerformance...
- Date_Dim
- Exam_Dim
- ExamResults_Fact
- HighSchoolType_Dim
- Lecturer_Dim
- Question_Dim
- Student_Dim

Queries

- CorrectnessPercenta...
- ExamSuccess_HighS...
- Passed_Students
- PASSED_TermAsses...
- PassedByDate
- PercentagePassedM...
- SuccessAttendence...

Defining extraction, transformation and loading process

The ETL (Extract, Transform, Load) process, which involved extracting data from the school operational database into the target database in the data warehouse, comprised the following steps:

Extracting:

- Connected to the source operational database.

- Retrieved the required data from the source database using SQL queries or other appropriate methods. In this case, the data was extracted from the dimension tables (Student_Dim, Course_Dim, Exam_Dim, Assessment_Dim, Date_Dim, Lecturer_Dim, and Question_Dim).

Transforming:

- Cleaned and validated the extracted data to ensure its quality and consistency. This involved handling missing values, removing duplicates, checking data types, and enforcing constraints.

- Applied data transformations based on the data warehouse's requirements, such as splitting or merging columns, converting data types, calculating derived values, or implementing business rules.

- For the initial load, loaded the data from each dimension table directly into the target database without major transformations. However, in future incremental loads or more complex scenarios, additional transformations might be required based on business requirements.

Loading:

- Connected to the target data warehouse database.

- Created the necessary tables in the target database if they did not exist. For each dimension table, created a corresponding table in the data warehouse.

- Loaded the transformed data from the source database into the target data warehouse tables using bulk loading or batch processing techniques for efficient data transfer.

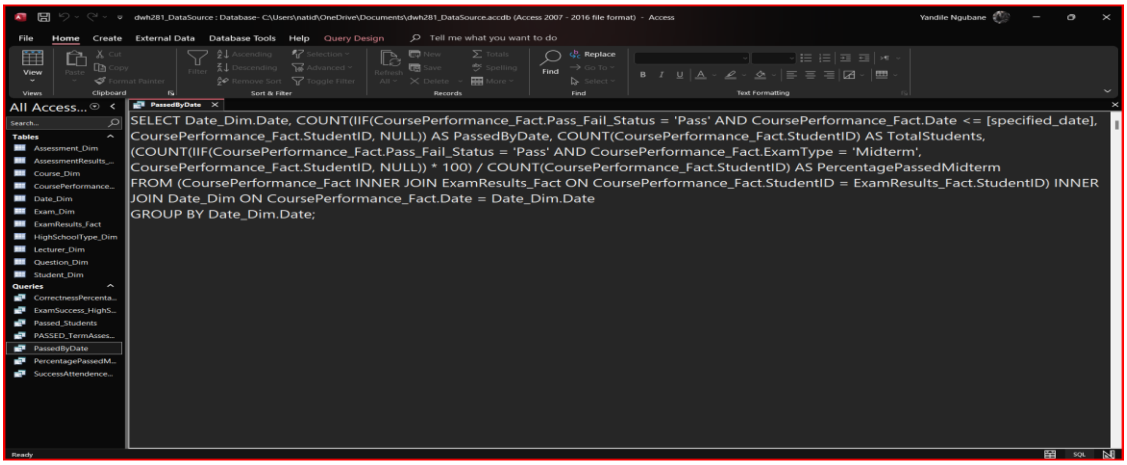
- Ensured that the data was loaded correctly into the target tables and handled any loading errors that occurred.

Validated:

- Performed post-load validation checks to confirm that the data had been successfully loaded into the target data warehouse. This included comparing the row counts between the source and target tables, checking data integrity, and verifying that all necessary transformations were applied correctly.

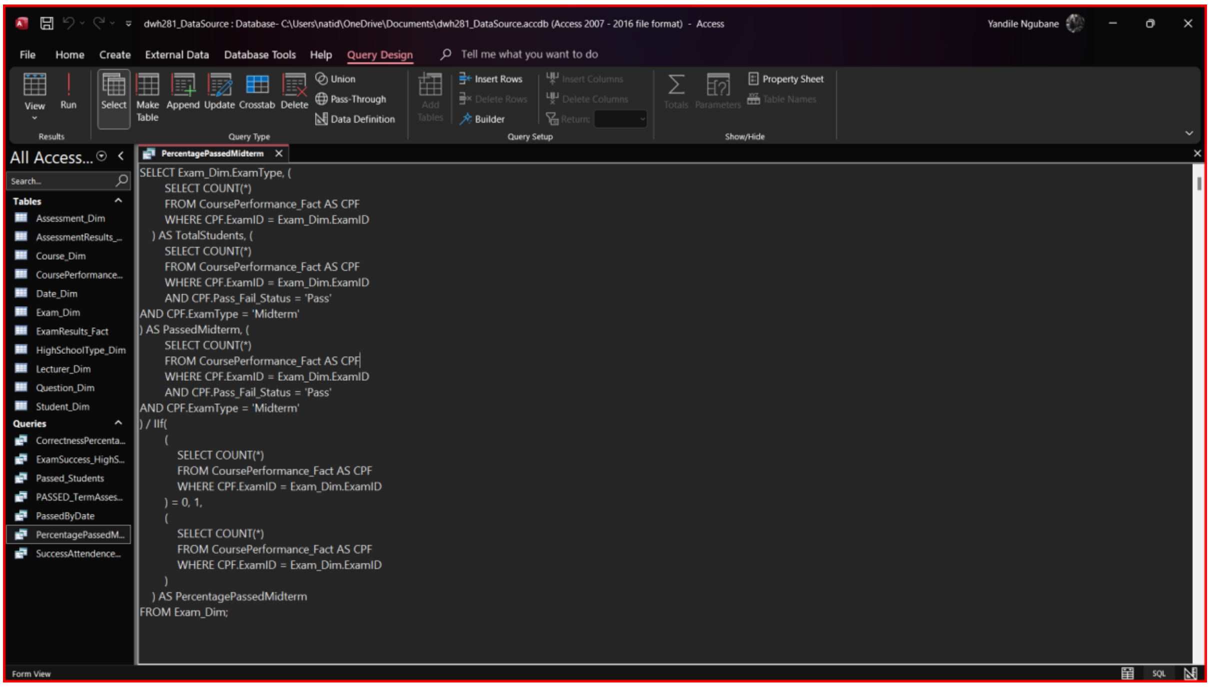
<u>User Request Queries</u>

1. Number of students having passed an exam by the specified date compared to their performance in course assessments and attendance of the lectures



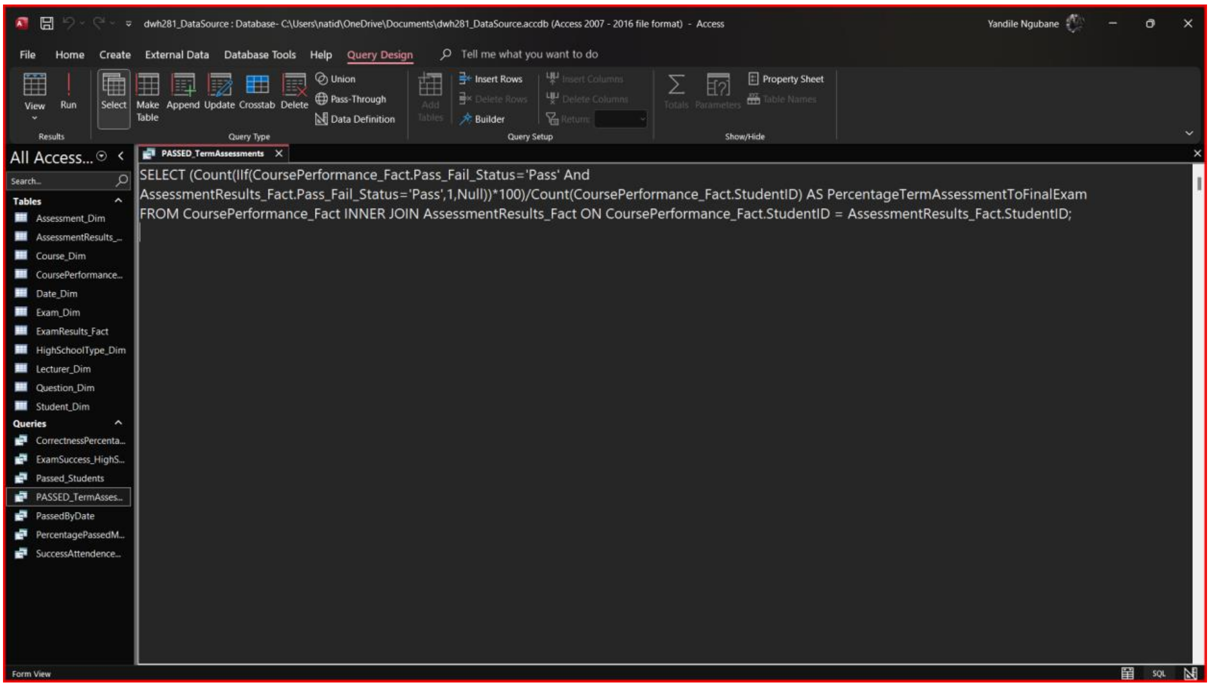
```
SELECT Date_Dim.Date, COUNT(IIF(CoursePerformance_Fact.Pass_Fail_Status = 'Pass' AND CoursePerformance_Fact.Date <= [specified_date],
CoursePerformance_Fact.StudentID, NULL)) AS PassedByDate, COUNT(CoursePerformance_Fact.StudentID) AS TotalStudents,
(COUNT(IIF(CoursePerformance_Fact.Pass_Fail_Status = 'Pass' AND CoursePerformance_Fact.ExamType = 'Midterm',
CoursePerformance_Fact.StudentID, NULL)) * 100) / COUNT(CoursePerformance_Fact.StudentID) AS PercentagePassedMidterm
FROM (CoursePerformance_Fact INNER JOIN ExamResults_Fact ON CoursePerformance_Fact.StudentID = ExamResults_Fact.StudentID) INNER
JOIN Date_Dim ON CoursePerformance_Fact.Date = Date_Dim.Date
GROUP BY Date_Dim.Date;
```

2. Percentage of students having passed the exam by taking the mid-term exams.



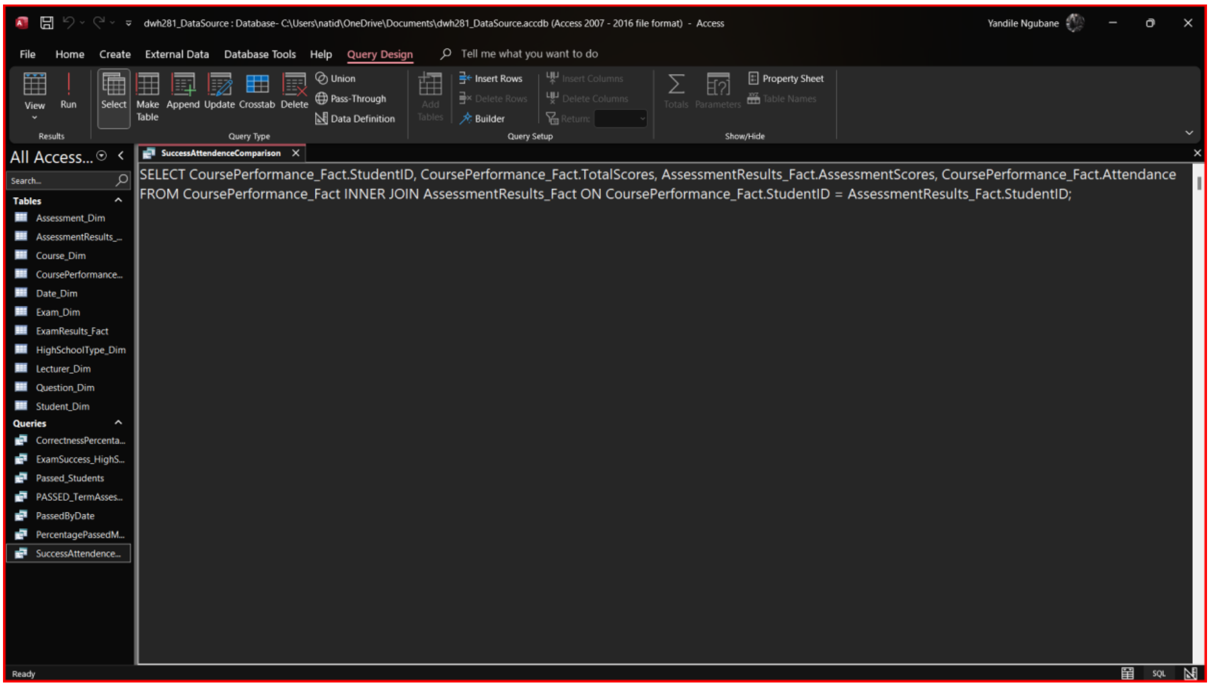
```sql
SELECT Exam_Dim.ExamType, (
      SELECT COUNT(*)
      FROM CoursePerformance_Fact AS CPF
      WHERE CPF.ExamID = Exam_Dim.ExamID
   ) AS TotalStudents, (
      SELECT COUNT(*)
      FROM CoursePerformance_Fact AS CPF
      WHERE CPF.ExamID = Exam_Dim.ExamID
      AND CPF.Pass_Fail_Status = 'Pass'
AND CPF.ExamType = 'Midterm'
) AS PassedMidterm, (
      SELECT COUNT(*)
      FROM CoursePerformance_Fact AS CPF
      WHERE CPF.ExamID = Exam_Dim.ExamID
      AND CPF.Pass_Fail_Status = 'Pass'
AND CPF.ExamType = 'Midterm'
) / IIf(
      (
         SELECT COUNT(*)
         FROM CoursePerformance_Fact AS CPF
         WHERE CPF.ExamID = Exam_Dim.ExamID
      ) = 0, 1,
      (
         SELECT COUNT(*)
         FROM CoursePerformance_Fact AS CPF
         WHERE CPF.ExamID = Exam_Dim.ExamID
      )
   ) AS PercentagePassedMidterm
FROM Exam_Dim;
```

3. Percentage of students whose term assessment scores allow them take final exam.



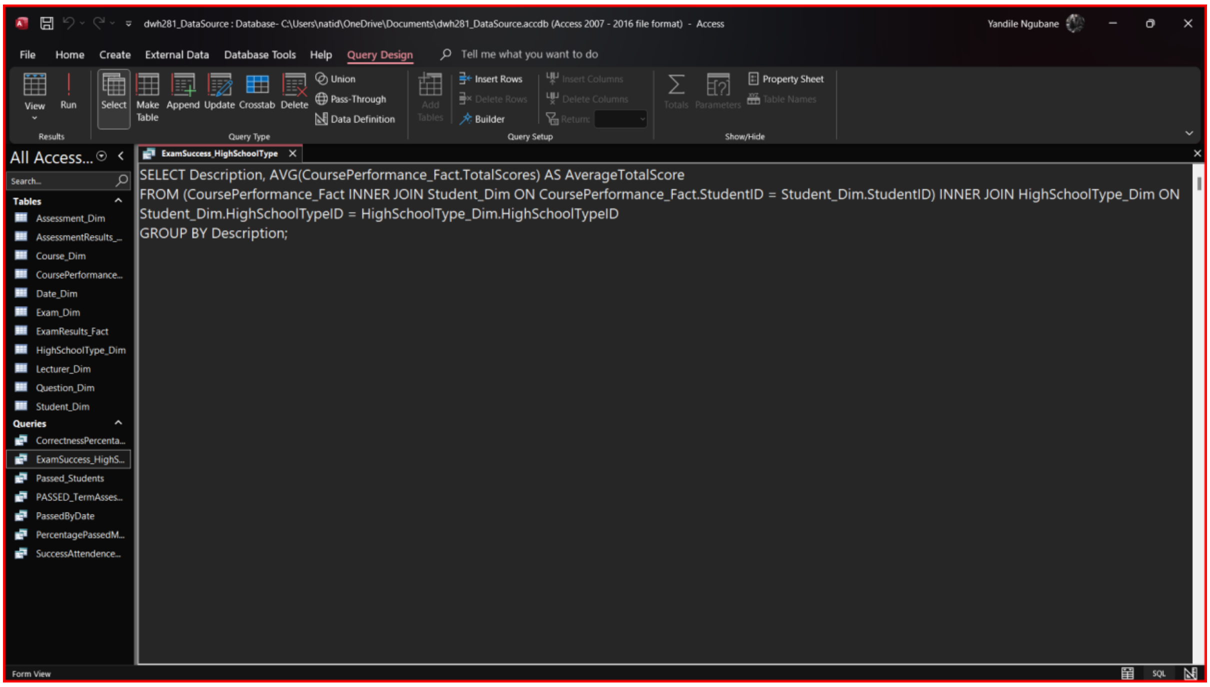
```sql
SELECT (Count(IIf(CoursePerformance_Fact.Pass_Fail_Status='Pass' And
AssessmentResults_Fact.Pass_Fail_Status='Pass',1,Null))*100)/Count(CoursePerformance_Fact.StudentID) AS PercentageTermAssessmentToFinalExam
FROM CoursePerformance_Fact INNER JOIN AssessmentResults_Fact ON CoursePerformance_Fact.StudentID = AssessmentResults_Fact.StudentID;
```

4. The grades that students got compared to their success in term assessment and attendance of the lectures.



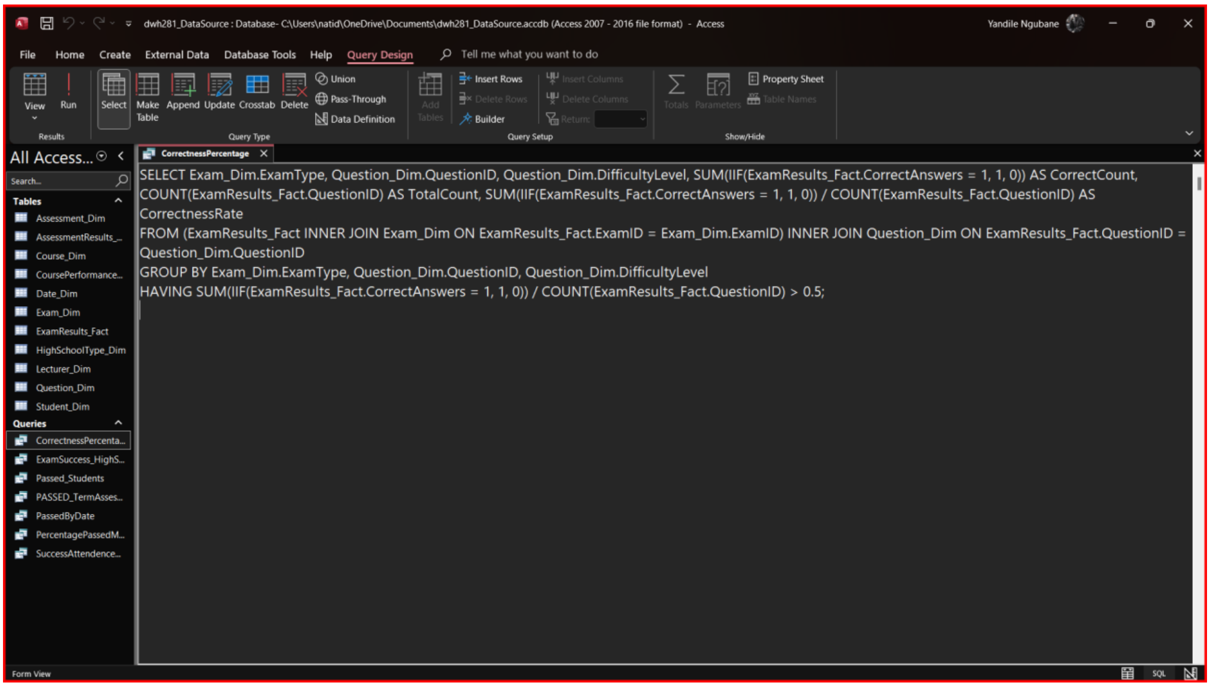
```
SELECT CoursePerformance_Fact.StudentID, CoursePerformance_Fact.TotalScores, AssessmentResults_Fact.AssessmentScores, CoursePerformance_Fact.Attendance
FROM CoursePerformance_Fact INNER JOIN AssessmentResults_Fact ON CoursePerformance_Fact.StudentID = AssessmentResults_Fact.StudentID;
```

5. Correlation between success on exams and different high school types the students attended.



```
SELECT Description, AVG(CoursePerformance_Fact.TotalScores) AS AverageTotalScore
FROM (CoursePerformance_Fact INNER JOIN Student_Dim ON CoursePerformance_Fact.StudentID = Student_Dim.StudentID) INNER JOIN HighSchoolType_Dim ON Student_Dim.HighSchoolTypeID = HighSchoolType_Dim.HighSchoolTypeID
GROUP BY Description;
```

6. Correctness of the answers to the questions on the exams.



```
SELECT Exam_Dim.ExamType, Question_Dim.QuestionID, Question_Dim.DifficultyLevel, SUM(IIF(ExamResults_Fact.CorrectAnswers = 1, 1, 0)) AS CorrectCount,
COUNT(ExamResults_Fact.QuestionID) AS TotalCount, SUM(IIF(ExamResults_Fact.CorrectAnswers = 1, 1, 0)) / COUNT(ExamResults_Fact.QuestionID) AS
CorrectnessRate
FROM (ExamResults_Fact INNER JOIN Exam_Dim ON ExamResults_Fact.ExamID = Exam_Dim.ExamID) INNER JOIN Question_Dim ON ExamResults_Fact.QuestionID =
Question_Dim.QuestionID
GROUP BY Exam_Dim.ExamType, Question_Dim.QuestionID, Question_Dim.DifficultyLevel
HAVING SUM(IIF(ExamResults_Fact.CorrectAnswers = 1, 1, 0)) / COUNT(ExamResults_Fact.QuestionID) > 0.5;
```

Extraction, transformation and loading process scripts

Loading/Transformation Scripts(SQL)

Student_Dim

```
INSERT INTO Student_Dim (Name, HighSchoolType)
SELECT Name, HighSchoolType
FROM Student_Source;
```

Course_Dim

```
INSERT INTO Course_Dim (CourseName, Credits)
SELECT CourseName, Credits
FROM Course_Source;
```

Exam_Dim

```
INSERT INTO Exam_Dim (ExamType, Date)
SELECT ExamType, Date
FROM Exam_Source;
```

Assessment_Dim

```
INSERT INTO Assessment_Dim (AssessmentType, Date)
SELECT AssessmentType, Date
FROM Assessment_Source;
```

Date_Dim

```
INSERT INTO Date_Dim (Date, Semester, Year)
SELECT Date, Semester, Year
FROM Date_Source;
```

Lecturer_Dim

```
INSERT INTO Lecturer_Dim (Name, Department)
SELECT Name, Department
FROM Lecturer_Source;
```

Question_Dim

```
INSERT INTO Question_Dim (DifficultyLevel, CorrectAnswer)
SELECT DifficultyLevel, CorrectAnswer
FROM Question_Source;
```

CoursePerformance_Fact

```
INSERT INTO CoursePerformance_Fact (StudentID, CourseID, Date,
ExamScores, Attendance, WeeklyAssessmentScores, TotalScores,
Pass_Fail_Status)
SELECT s.StudentID, c.CourseID, cp.Date, cp.ExamScores, cp.Attendance,
cp.WeeklyAssessmentScores, cp.TotalScores, cp.Pass_Fail_Status
FROM CoursePerformance_Source cp
(INNER JOIN Student_Source s ON cp.StudentID = s.StudentID)
```

```
INNER JOIN Course_Source c ON cp.CourseID = c.CourseID;
```

ExamResults_Fact

```
INSERT INTO ExamResults_Fact (StudentID, ExamID, ExamScores,
CorrectAnswers, DifficultyLevel, AverageScores, Pass_Fail_Status)
SELECT s.StudentID, e.ExamID, er.ExamScores, er.CorrectAnswers,
er.DifficultyLevel, er.AverageScores, er.Pass_Fail_Status
FROM ExamResults_Source er
(INNER JOIN Student_Source s ON er.StudentID = s.StudentID)
INNER JOIN Exam_Source e ON er.ExamID = e.ExamID;
```

AssessmentResults_Fact

```
INSERT INTO AssessmentResults_Fact (StudentID, AssessmentID,
AssessmentScores, CorrectAnswers, AverageScores, Pass_Fail_Status)
SELECT Student_Dim.StudentID, Assessment_Dim.AssessmentID,
AssessmentResults_Facts.AssessmentScores,
AssessmentResults_Facts.CorrectAnswers,
AssessmentResults_Facts.AverageScores,
AssessmentResults_Facts.Pass_Fail_Status
FROM AssessmentResults_Source ar
(INNER JOIN Student_Source Student_Dim ON
AssessmentResults_Facts.StudentID = s.StudentID)
INNER JOIN Assessment_Source a ON ar.AssessmentID = a.AssessmentID;
```

Loading Scripts(SQL)

The loading SQL Scripts are based on the assumption that they was a middle temporary table that the extracted and transformed data was loaded into.

Date_Dim

```
INSERT INTO Date_Dim (Date, Semester, Year)
SELECT Date, Semester, Year
FROM Transformed_DateData;
```

Lecturer_Dim

```
INSERT INTO Lecturer_Dim (Name, Department)
SELECT Name, Department
FROM Transformed_LecturerData;
```

Question_Dim

```
INSERT INTO Question_Dim (DifficultyLevel, CorrectAnswer)
SELECT DifficultyLevel, CorrectAnswer
FROM Transformed_QuestionData;
```

Course_Dim

```
INSERT INTO Course_Dim (CourseName, Credits)
SELECT CourseName, Credits
FROM Transformed_CourseData;
```

Assessment_Dim

```
INSERT INTO Assessment_Dim (AssessmentType, Date)
SELECT AssessmentType, Date
FROM Transformed_AssessmentData;
```

Exam_Dim

```
INSERT INTO Exam_Dim (ExamType, Date)
SELECT ExamType, Date
FROM Transformed_ExamData;
```

Student_Dim

```
INSERT INTO Student_Dim (Name, HighSchoolType)
SELECT Name, HighSchoolType
FROM Transformed_StudentData;
```

CoursePerformance_Fact

```
INSERT INTO CoursePerformance_Fact (StudentID, CourseID, Date,
ExamScores, Attendance, WeeklyAssessmentScores, TotalScores,
Pass_Fail_Status)
SELECT StudentID, CourseID, Date, ExamScores, Attendance,
WeeklyAssessmentScores, TotalScores, Pass_Fail_Status
FROM Transformed_CoursePerformanceData;
```

ExamResults_Fact

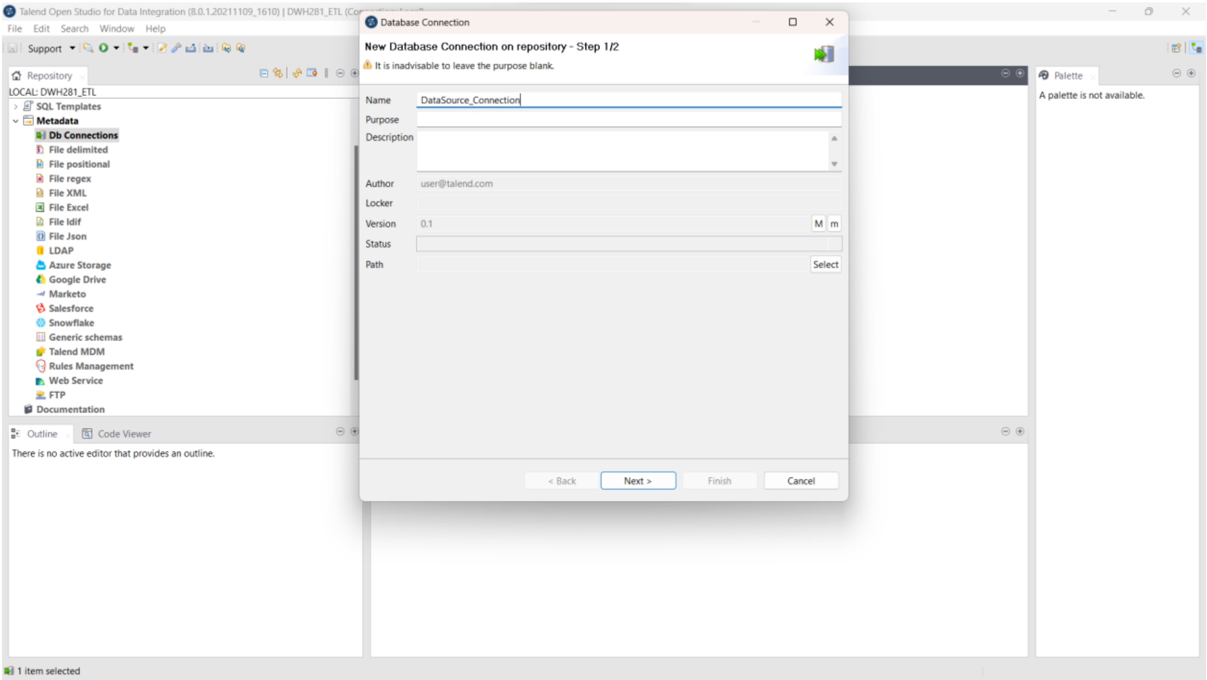
```
INSERT INTO ExamResults_Fact (StudentID, ExamID, ExamScores,
CorrectAnswers, DifficultyLevel, AverageScores, Pass_Fail_Status)
SELECT StudentID, ExamID, ExamScores, CorrectAnswers, DifficultyLevel,
AverageScores, Pass_Fail_Status
FROM Transformed_ExamResultsData;
```

AssessmentResults_Fact

```
INSERT INTO AssessmentResults_Fact (StudentID, AssessmentID,
AssessmentScores, CorrectAnswers, AverageScores, Pass_Fail_Status)
SELECT StudentID, AssessmentID, AssessmentScores, CorrectAnswers,
AverageScores, Pass_Fail_Status FROM Transformed_AssessmentResultsData;
```

Initial Load of the Datawarehouse

1. Connecting the Data Source Database to extract from:



New Database Connection on repository - Step 1/2

It is inadvisable to leave the purpose blank.

Name: DataSource_Connection

Purpose:

Description:

Author: user@talend.com

Locker:

Version: 0.1

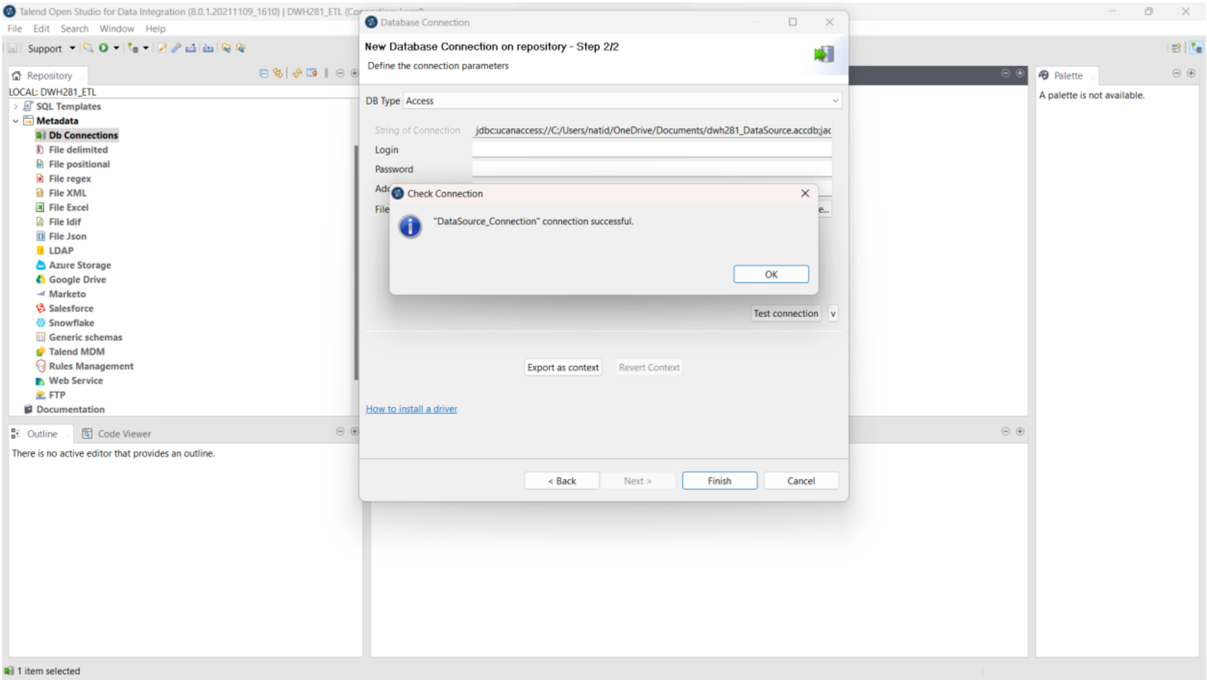
Status:

Path:

There is no active editor that provides an outline.

A palette is not available.

1 item selected



New Database Connection on repository - Step 2/2

Define the connection parameters

DB Type: Access

String of Connection: jdbc:ucanaccess://C:/Users/natid/OneDrive/Documents/dwh281_DataSource.accdb;jac

Login:

Password:

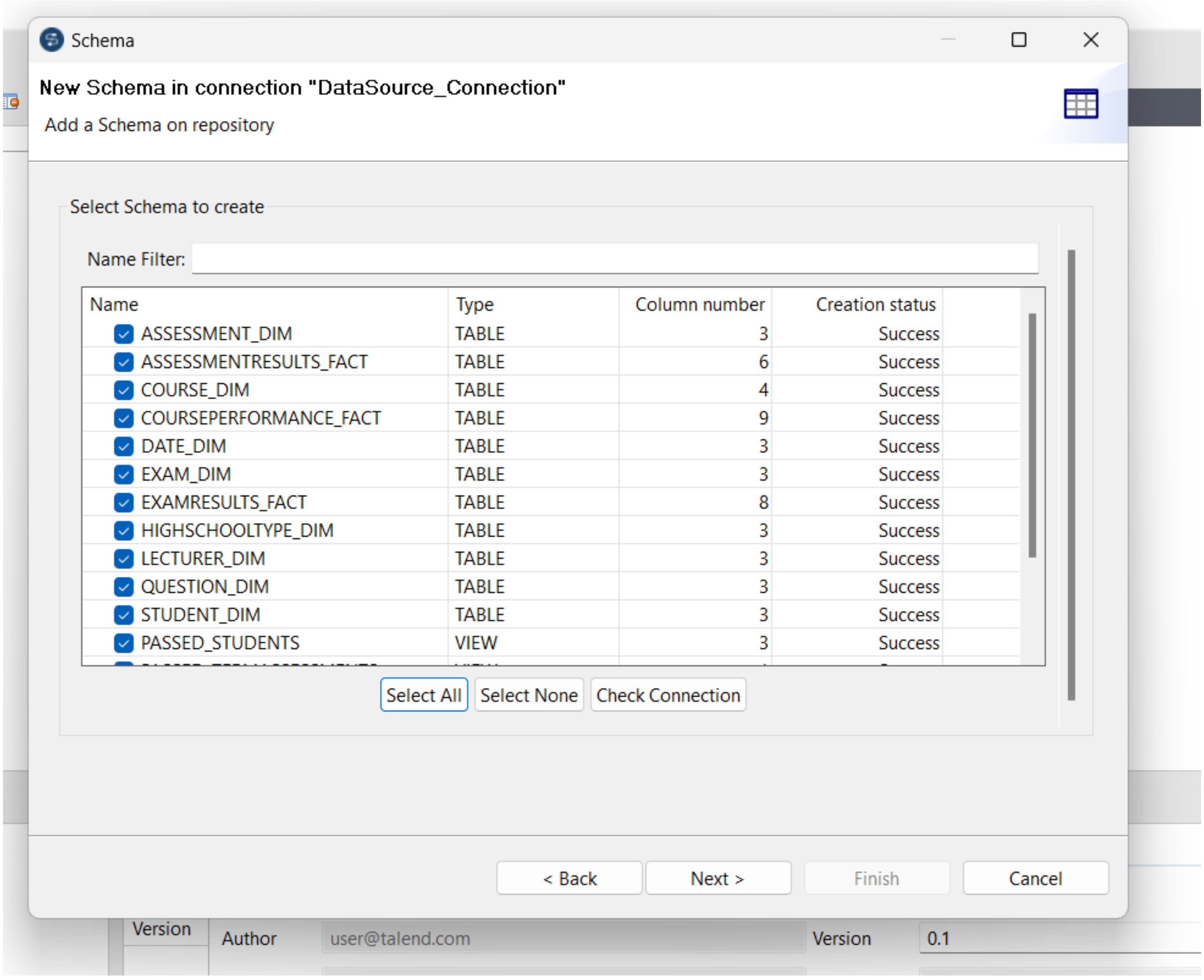
"DataSource_Connection" connection successful.

There is no active editor that provides an outline.

A palette is not available.

1 item selected

2. Retrieving Schema



New Schema in connection "DataSource_Connection"

Add a Schema on repository

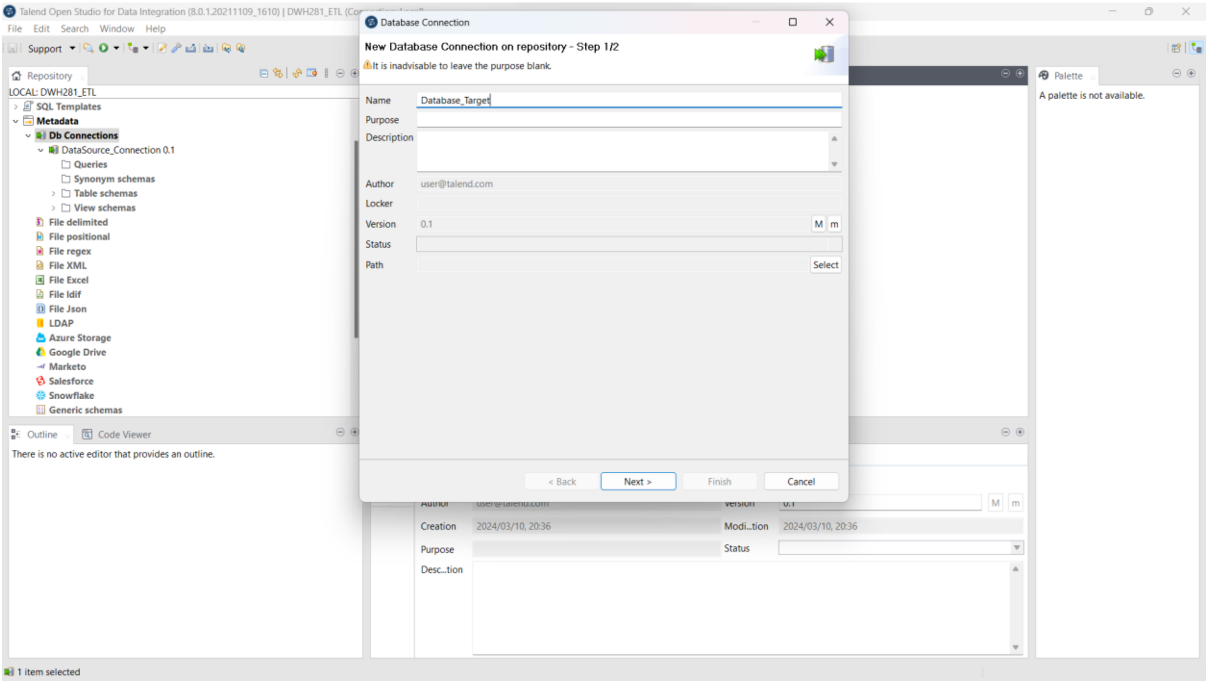
Select Schema to create

Name Filter:

Name	Type	Column number	Creation status
☑ ASSESSMENT_DIM	TABLE	3	Success
☑ ASSESSMENTRESULTS_FACT	TABLE	6	Success
☑ COURSE_DIM	TABLE	4	Success
☑ COURSEPERFORMANCE_FACT	TABLE	9	Success
☑ DATE_DIM	TABLE	3	Success
☑ EXAM_DIM	TABLE	3	Success
☑ EXAMRESULTS_FACT	TABLE	8	Success
☑ HIGHSCHOOLTYPE_DIM	TABLE	3	Success
☑ LECTURER_DIM	TABLE	3	Success
☑ QUESTION_DIM	TABLE	3	Success
☑ STUDENT_DIM	TABLE	3	Success
☑ PASSED_STUDENTS	VIEW	3	Success

Author user@talend.com Version 0.1

3. Connecting the Target Database



New Database Connection on repository - Step 1/2

⚠It is inadvisable to leave the purpose blank.

Name: Database_Target

Purpose:

Description:

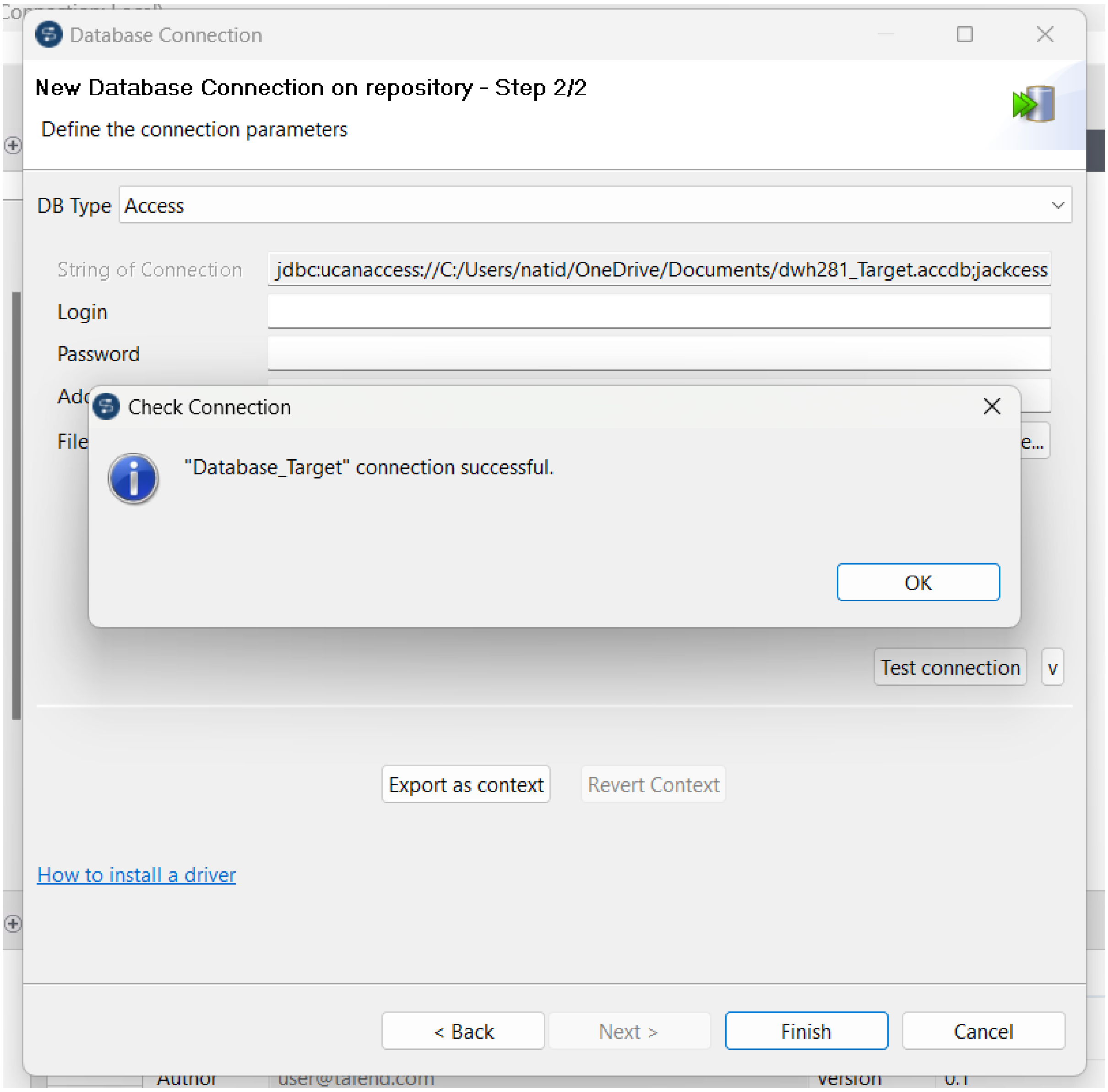
Author: user@talend.com

Locker:

Version: 0.1

Status:

Path:



Database Connection

New Database Connection on repository - Step 2/2

Define the connection parameters

DB Type: Access

String of Connection: jdbc:ucanaccess://C:/Users/natid/OneDrive/Documents/dwh281_Target.accdb;jackcess

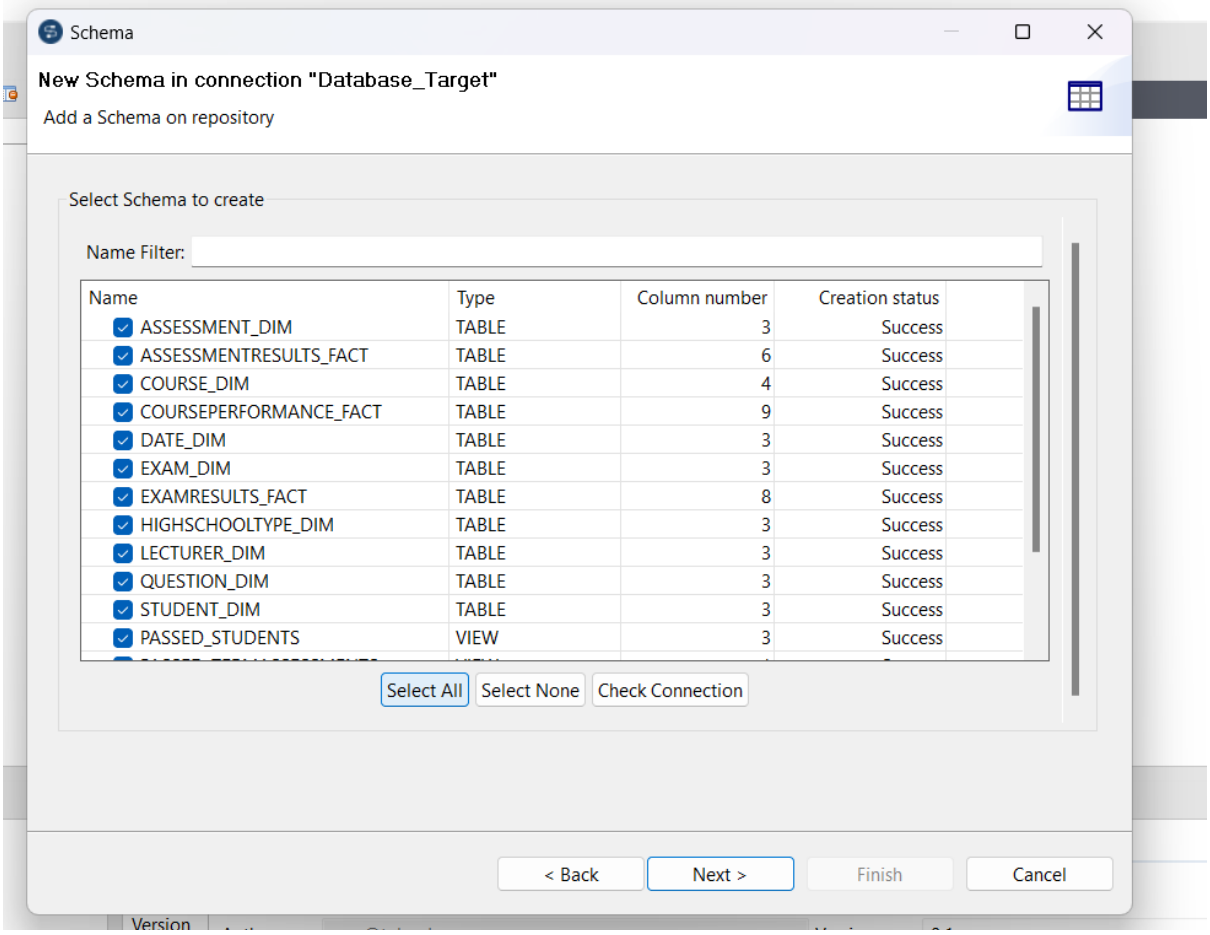
Login:

Password:

Check Connection

"Database_Target" connection successful.

4. Retrieving Schema



New Schema in connection "Database_Target"

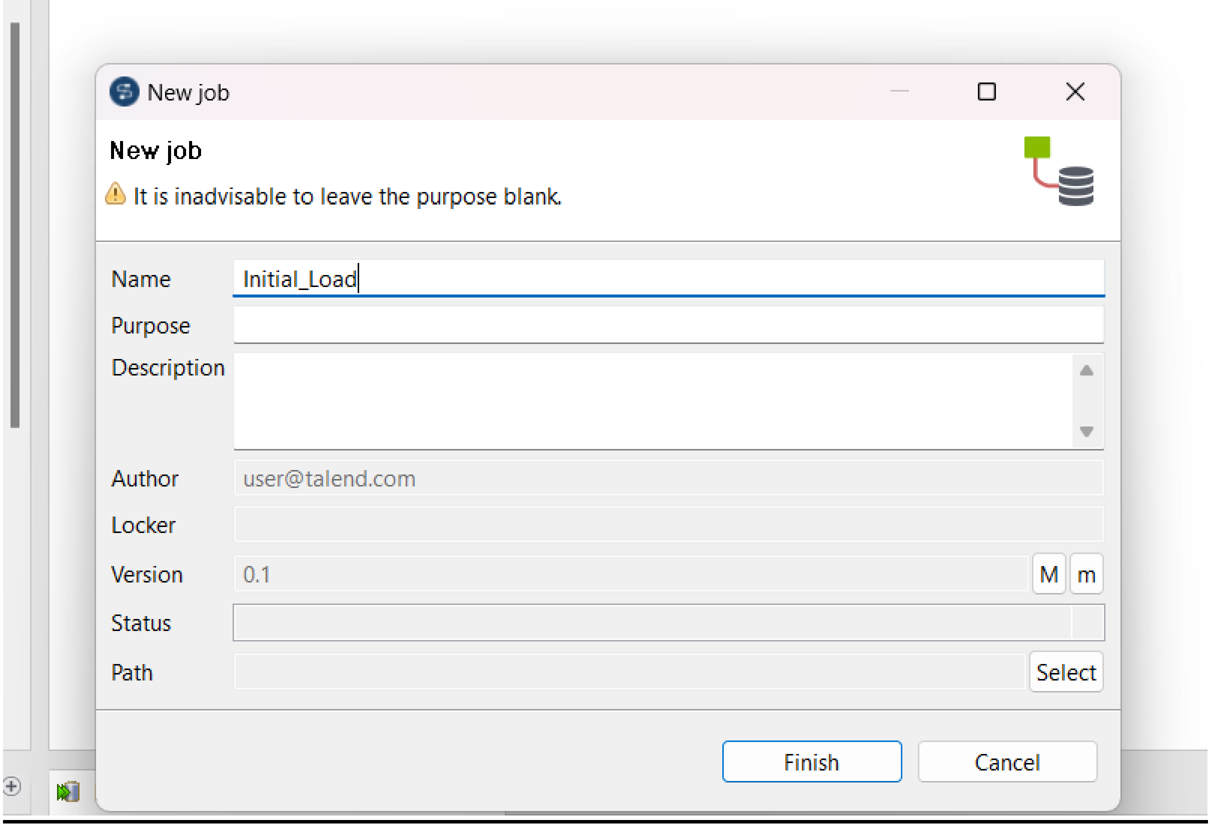
Add a Schema on repository

Select Schema to create

Name Filter:

Name	Type	Column number	Creation status
☑ ASSESSMENT_DIM	TABLE	3	Success
☑ ASSESSMENTRESULTS_FACT	TABLE	6	Success
☑ COURSE_DIM	TABLE	4	Success
☑ COURSEPERFORMANCE_FACT	TABLE	9	Success
☑ DATE_DIM	TABLE	3	Success
☑ EXAM_DIM	TABLE	3	Success
☑ EXAMRESULTS_FACT	TABLE	8	Success
☑ HIGHSCHOOLTYPE_DIM	TABLE	3	Success
☑ LECTURER_DIM	TABLE	3	Success
☑ QUESTION_DIM	TABLE	3	Success
☑ STUDENT_DIM	TABLE	3	Success
☑ PASSED_STUDENTS	VIEW	3	Success

Creating Initial Load Job



New job

New job

⚠ It is inadvisable to leave the purpose blank.

Name: Initial_Load

Purpose:

Description:

Author: user@talend.com

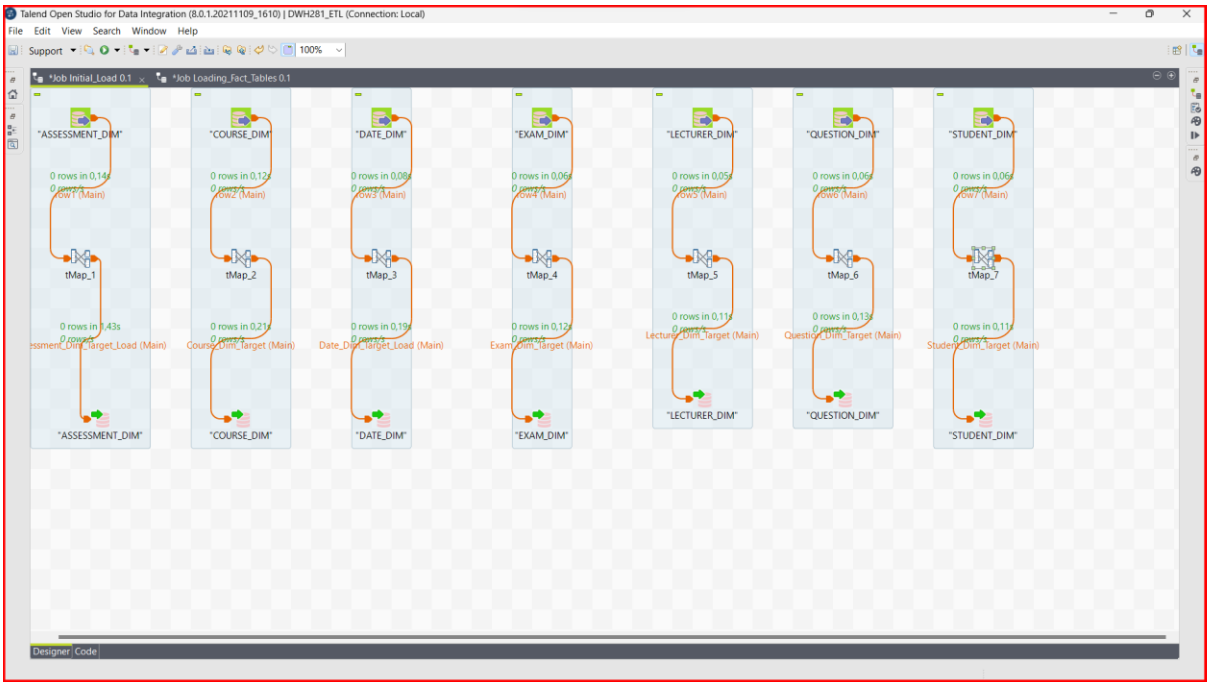
Locker:

Version: 0.1

Status:

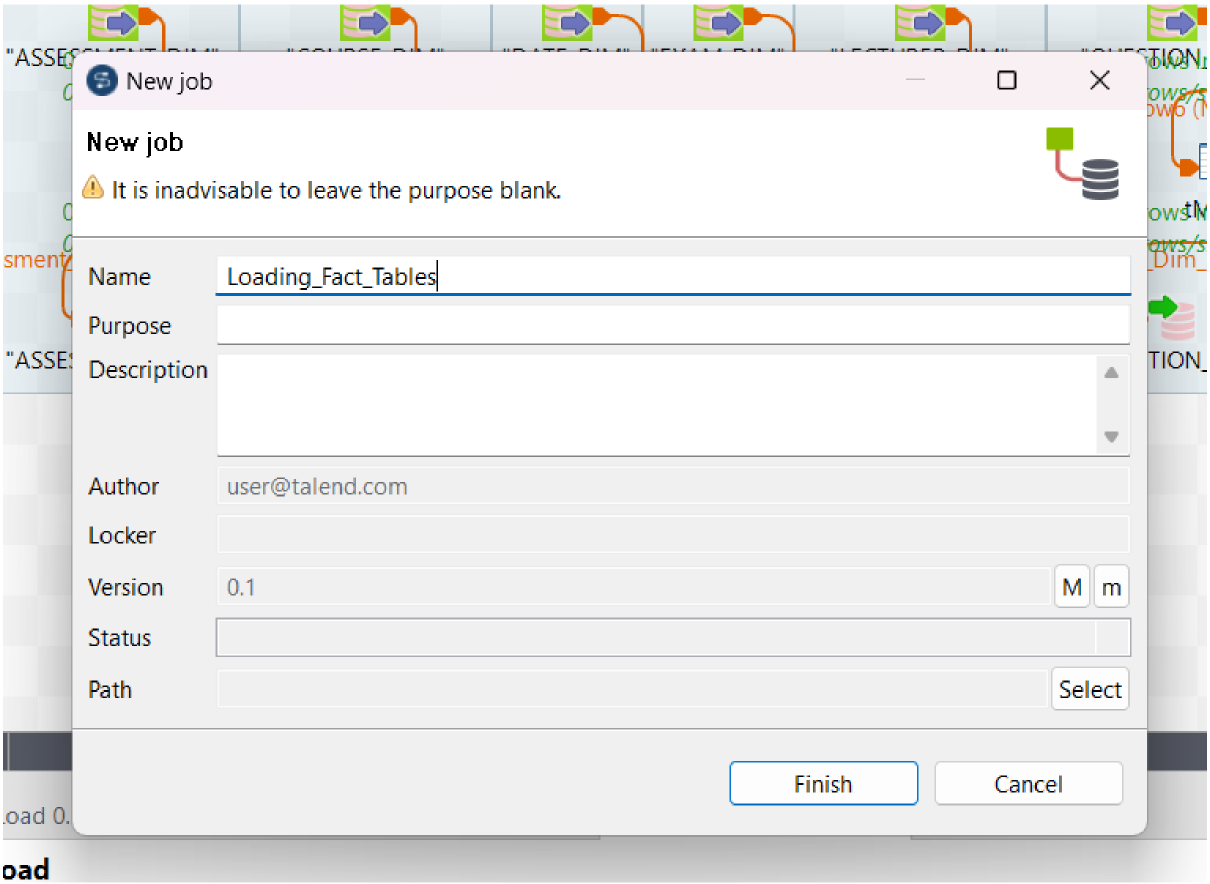
Path:

1. Loading Target Dimension Tables



Talend Open Studio for Data Integration (8.0.1.20211109_1610) | DWH281_ETL (Connection: Local)

2. Loading Fact Tables



New job

New job

⚠ It is inadvisable to leave the purpose blank.

Name: Loading_Fact_Tables

Purpose:

Description:

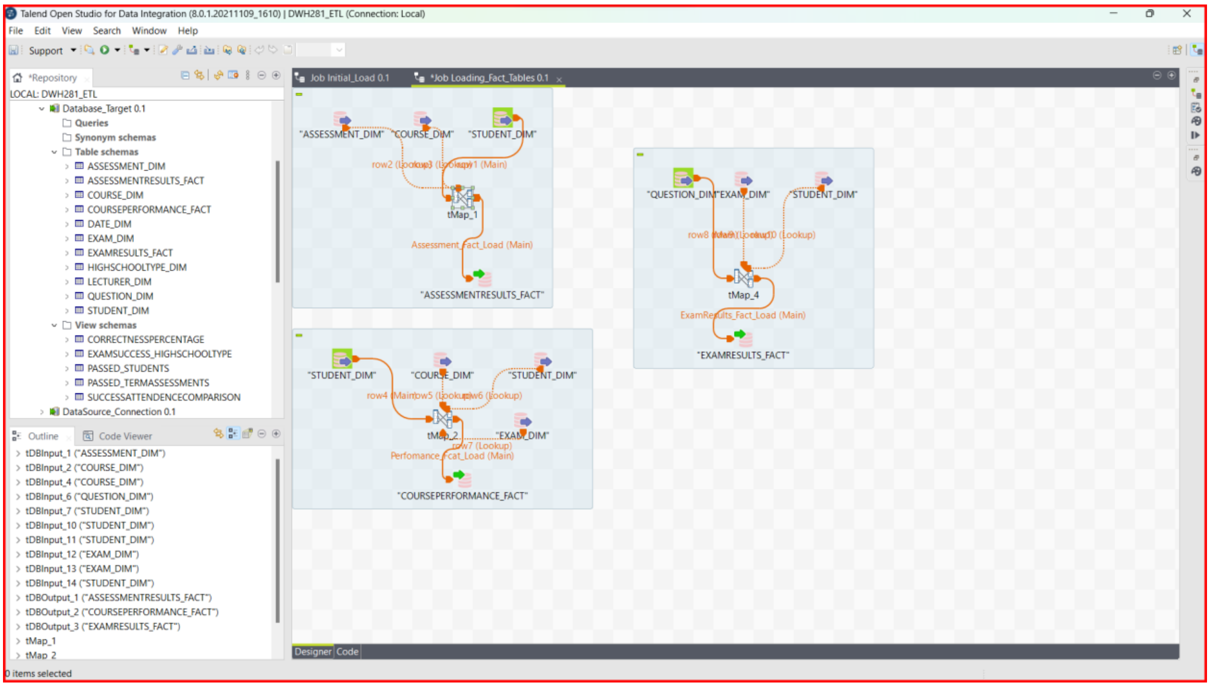
Author: user@talend.com

Locker:

Version: 0.1

Status:

Path:



Talend Open Studio for Data Integration (8.0.1.20211109_1610) | DWH281_ETL (Connection: Local)

*Repository

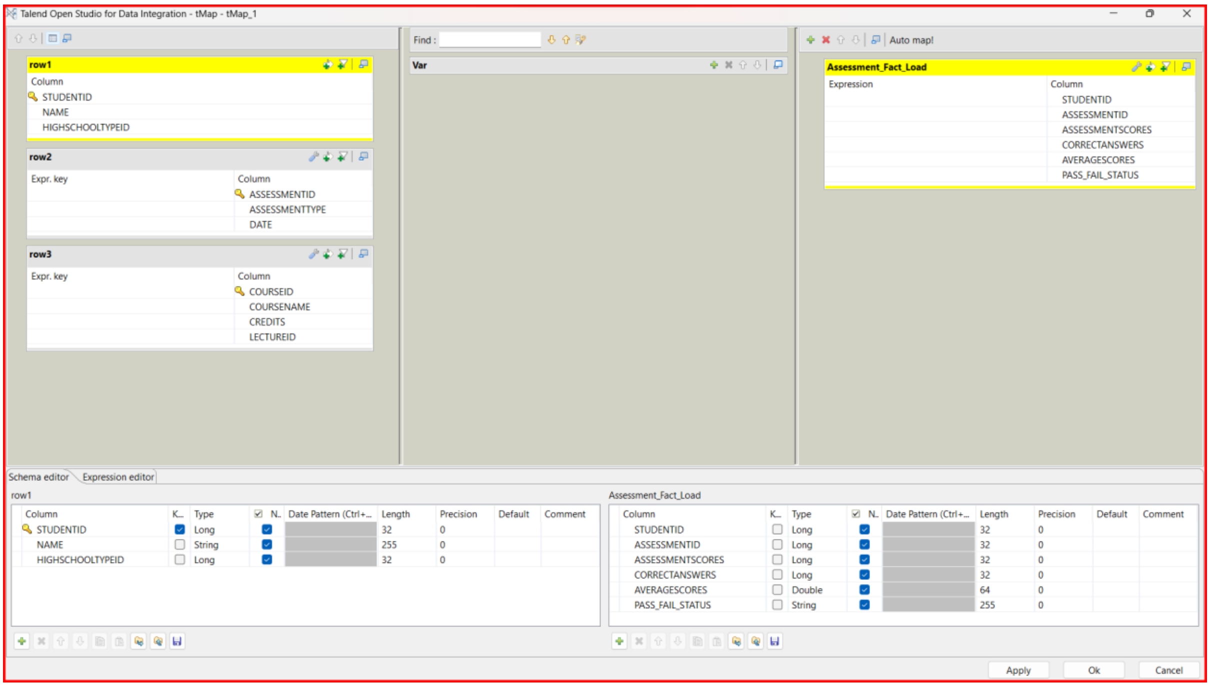
LOCAL: DWH281_ETL

- Database_Target 0.1
  - Queries
  - Synonym schemas
  - Table schemas
    - ASSESSMENT_DIM
    - ASSESSMENTRESULTS_FACT
    - COURSE_DIM
    - COURSEPERFORMANCE_FACT
    - DATE_DIM
    - EXAM_DIM
    - EXAMRESULTS_FACT
    - HIGHSCHOOLTYPE_DIM
    - LECTURER_DIM
    - QUESTION_DIM
    - STUDENT_DIM
  - View schemas
    - CORRECTNESSPERCENTAGE
    - EXAMSUCCESS_HIGHSCHOOLTYPE
    - PASSED_STUDENTS
    - PASSED_TERMASSESSMENTS
    - SUCCESSATTENDENCECOMPARISON
- DataSource_Connection 0.1

Outline

- tDBInput_1 ("ASSESSMENT_DIM")
- tDBInput_2 ("COURSE_DIM")
- tDBInput_4 ("COURSE_DIM")
- tDBInput_6 ("QUESTION_DIM")
- tDBInput_7 ("STUDENT_DIM")
- tDBInput_10 ("STUDENT_DIM")
- tDBInput_11 ("STUDENT_DIM")
- tDBInput_12 ("EXAM_DIM")
- tDBInput_13 ("EXAM_DIM")
- tDBInput_14 ("STUDENT_DIM")
- tDBOutput_1 ("ASSESSMENTRESULTS_FACT")
- tDBOutput_2 ("COURSEPERFORMANCE_FACT")
- tDBOutput_3 ("EXAMRESULTS_FACT")
- tMap_1
- tMap_2

0 items selected



Talend Open Studio for Data Integration - tMap - tMap_1

Find :

Auto map!

row1

Column
STUDENTID
NAME
HIGHSCHOOLTYPEID

row2

Expr. key	Column
	ASSESSMENTID
	ASSESSMENTTYPE
	DATE

row3

Expr. key	Column
	COURSEID
	COURSENAME
	CREDITS
	LECTUREID

Var

Assessment_Fact_Load

Expression	Column
	STUDENTID
	ASSESSMENTID
	ASSESSMENTSCORES
	CORRECTANSWERS
	AVERAGESCORES
	PASS_FAIL_STATUS

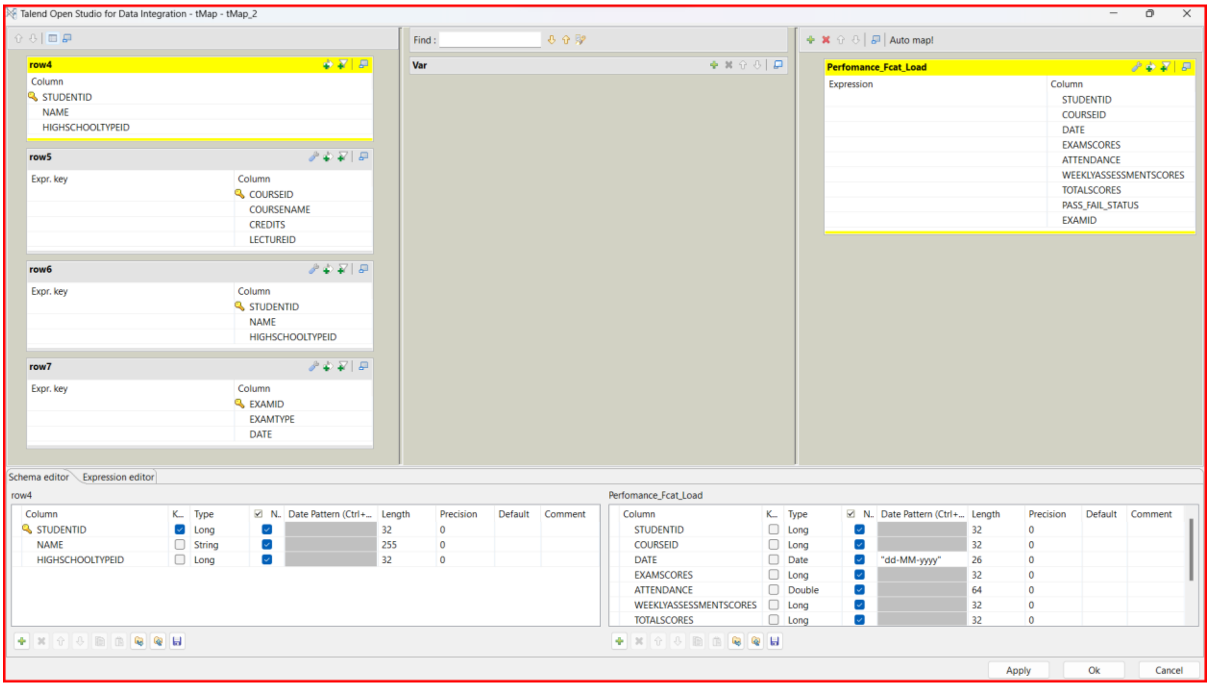
Schema editor | Expression editor

row1

Column	K..	Type	☑ N..	Date Pattern (Ctrl+...	Length	Precision	Default	Comment
STUDENTID	☑	Long	☑		32	0		
NAME	☐	String	☑		255	0		
HIGHSCHOOLTYPEID	☐	Long	☑		32	0		

Assessment_Fact_Load

Column	K..	Type	☑ N..	Date Pattern (Ctrl+...	Length	Precision	Default	Comment
STUDENTID	☐	Long	☑		32	0		
ASSESSMENTID	☐	Long	☑		32	0		
ASSESSMENTSCORES	☐	Long	☑		32	0		
CORRECTANSWERS	☐	Long	☑		32	0		
AVERAGESCORES	☐	Double	☑		64	0		
PASS_FAIL_STATUS	☐	String	☑		255	0		



Talend Open Studio for Data Integration - tMap - tMap_2

Find :

Auto map!

row4

Column
STUDENTID
NAME
HIGHSCHOOLTYPEID

row5

Expr. key	Column
	COURSEID
	COURSENAME
	CREDITS
	LECTUREID

row6

Expr. key	Column
	STUDENTID
	NAME
	HIGHSCHOOLTYPEID

row7

Expr. key	Column
	EXAMID
	EXAMTYPE
	DATE

Var

Perfomance_Fcat_Load

Expression	Column
	STUDENTID
	COURSEID
	DATE
	EXAMSCORES
	ATTENDANCE
	WEEKLYASSESSMENTSCORES
	TOTALSCORES
	PASS_FAIL_STATUS
	EXAMID

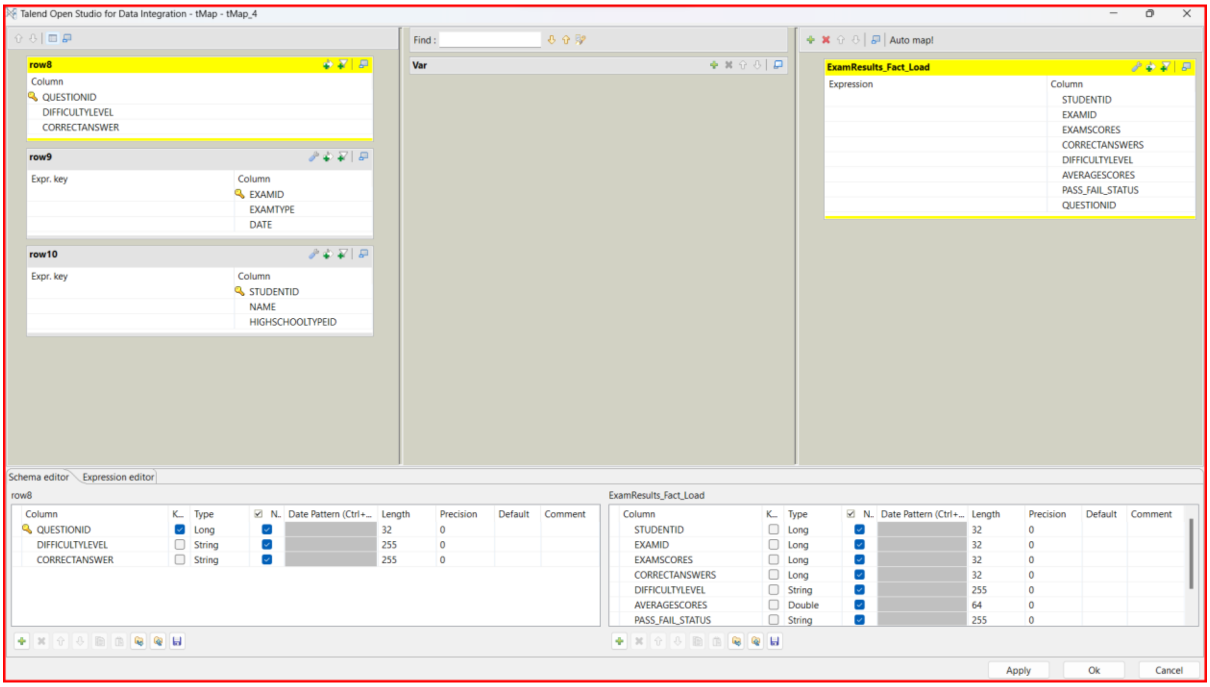
Schema editor | Expression editor

row4

Column	K..	Type	☑ N..	Date Pattern (Ctrl+...	Length	Precision	Default	Comment
STUDENTID	☑	Long	☑		32	0		
NAME	☐	String	☑		255	0		
HIGHSCHOOLTYPEID	☐	Long	☑		32	0		

Perfomance_Fcat_Load

Column	K..	Type	☑ N..	Date Pattern (Ctrl+...	Length	Precision	Default	Comment
STUDENTID	☐	Long	☑		32	0		
COURSEID	☐	Long	☑		32	0		
DATE	☐	Date	☑	"dd-MM-yyyy"	26	0		
EXAMSCORES	☐	Long	☑		32	0		
ATTENDANCE	☐	Double	☑		64	0		
WEEKLYASSESSMENTSCORES	☐	Long	☑		32	0		
TOTALSCORES	☐	Long	☑		32	0		



row8

Column
QUESTIONID
DIFFICULTYLEVEL
CORRECTANSWER

row9

Expr. key	Column
	EXAMID
	EXAMTYPE
	DATE

row10

Expr. key	Column
	STUDENTID
	NAME
	HIGHSCHOOLTYPEID

ExamResults_Fact_Load

Expression	Column
	STUDENTID
	EXAMID
	EXAMSCORES
	CORRECTANSWERS
	DIFFICULTYLEVEL
	AVERAGESCORES
	PASS_FAIL_STATUS
	QUESTIONID

row8

Column	K...	Type	☑ N..	Date Pattern (Ctrl+...	Length	Precision	Default	Comment
QUESTIONID	☑	Long	☑		32	0		
DIFFICULTYLEVEL	□	String	☑		255	0		
CORRECTANSWER	□	String	☑		255	0		

ExamResults_Fact_Load

Column	K...	Type	☑ N..	Date Pattern (Ctrl+...	Length	Precision	Default	Comment
STUDENTID	□	Long	☑		32	0		
EXAMID	□	Long	☑		32	0		
EXAMSCORES	□	Long	☑		32	0		
CORRECTANSWERS	□	Long	☑		32	0		
DIFFICULTYLEVEL	□	String	☑		255	0		
AVERAGESCORES	□	Double	☑		64	0		
PASS_FAIL_STATUS	□	String	☑		255	0		