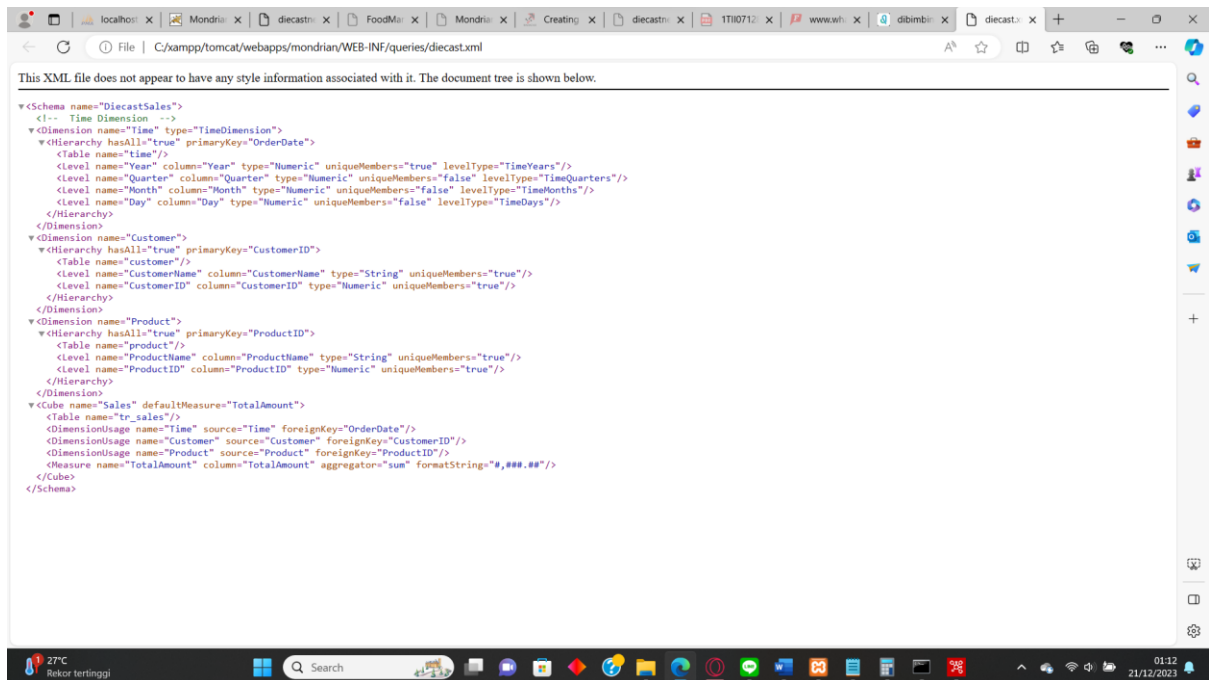


Julius Calvin Saputra (0000068626)

1. File XML



Time Dimension

Dimension name: "Time"

Dimension type: "TimeDimension"

Hierarchy: Year, Quarter, Month, and Day.

Table: Time table

Primary key: OrderDate

Customer dimension

Dimension name: "Customer"

Hierarchy: CustomerName and CustomerID levels.

Table: Customer

Primary key: CustomerID

Product dimension

Dimension name: "Product"

Hierarchy: ProductName and ProductID levels.

Table: Product table

Primary key: ProductID

Sales cube

Cube name: "Sales"

Measure: Total Amount, where it uses the aggregator function "sum" to calculate the total.

Table: tr_sales

Dimension Usage: Connects cubes with time, customer \, and product dimensions through foreign key

2. File JSP

```
diecast.jsp
File Edit View

<%@ page session="true" contentType="text/html; charset=ISO-8859-1" %>
<%@ taglib uri="http://www.tonbeller.com/jpivot" prefix="jp" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jstl/core" %>

<jp:mondrianQuery id="query01"
  jdbcDriver="com.mysql.jdbc.Driver"
  jdbcUrl="jdbc:mysql://localhost/diecastnew?user=root&password="
  catalogUri="/WEB-INF/queries/diecast.xml">

  SELECT
    {[Measures].[TotalAmount]} ON COLUMNS,
    {[Customer].[CustomerID].Members} * {[Product].[ProductID].Members, [Product].[ProductName].Members} ON ROWS
  FROM [Sales]

</jp:mondrianQuery>

<c:set var="title01" scope="session">die cast</c:set>
```

The combination of CustomerID, ProductID, and ProductName in the row of the Sales cube, along with the selection of TotalAmount in the column, constitute the performed query.

3. Output

		Measures
Customer	Product	TotalAmount
3	116	
	119	
	118	
	115	
	120	
	117	
	-Bburago Bugatti Chiron	
	-Hotwheels Aston Martin Vantage	
	-Jada Chevrolet Camaro	
	-Maisto Ford GT	
1	116	
	119	
	118	
	115	
	120	
	117	
	-Bburago Bugatti Chiron	
	-Hotwheels Aston Martin Vantage	
	-Jada Chevrolet Camaro	
	-Maisto Ford GT	
4	116	
	119	
	118	
	115	
	120	
	117	
	-Bburago Bugatti Chiron	
	-Hotwheels Aston Martin Vantage	
	-Jada Chevrolet Camaro	
	-Maisto Ford GT	

4	-Welly Mercedes-Benz GLE	
	116	
	119	
	118	
	115	
	120	
	117	
	-Bburago Bugatti Chiron	
	-Hotwheels Aston Martin Vantage	
	-Jada Chevrolet Camaro	
2	116	
	119	
	118	
	115	
	120	
	117	
	-Bburago Bugatti Chiron	
	-Hotwheels Aston Martin Vantage	
	-Jada Chevrolet Camaro	
	-Maisto Ford GT	

● Julius.1.Maisto Ford GT.115. ● Julius.1.Maisto Toyota Land Cruiser.120.
 ● Julius.1.Welly Mercedes-Benz GLE.117. ● Julius.1.Bburago Bugatti Chiron.
 ● Julius.1.Hotwheels Aston Martin Vantage. ● Julius.1.Jada Chevrolet Camaro.
 ● Julius.1.Maisto Ford GT. ● Julius.1.Maisto Toyota Land Cruiser.
 ● Julius.1.Welly Mercedes-Benz GLE. ● N/A.4.Bburago Bugatti Chiron.116.
 ● N/A.4.Hotwheels Aston Martin Vantage.119. ● N/A.4.Jada Chevrolet Camaro.118.
 ● N/A.4.Maisto Ford GT.115. ● N/A.4.Maisto Toyota Land Cruiser.120.
 ● N/A.4.Welly Mercedes-Benz GLE.117. ● N/A.4.Bburago Bugatti Chiron.
 ● N/A.4.Hotwheels Aston Martin Vantage. ● N/A.4.Jada Chevrolet Camaro. ● N/A.4.Maisto Ford GT.
 ● N/A.4.Maisto Toyota Land Cruiser. ● N/A.4.Welly Mercedes-Benz GLE.
 ● Vinh.2.Bburago Bugatti Chiron.116. ● Vinh.2.Hotwheels Aston Martin Vantage.119.
 ● Vinh.2.Jada Chevrolet Camaro.118. ● Vinh.2.Maisto Ford GT.115.
 ● Vinh.2.Maisto Toyota Land Cruiser.120. ● Vinh.2.Welly Mercedes-Benz GLE.117.
 ● Vinh.2.Bburago Bugatti Chiron. ● Vinh.2.Hotwheels Aston Martin Vantage.
 ● Vinh.2.Jada Chevrolet Camaro. ● Vinh.2.Maisto Ford GT. ● Vinh.2.Maisto Toyota Land Cruiser.
 ● Vinh.2.Welly Mercedes-Benz GLE.

OLEB Cube menunjukkan setiap pelanggan dan apa yang mereka beli, misalnya Julius membeli 1 Maisto Ford GT. Ini berarti perusahaan dapat melacak penjualan dan melihat apa yang dibeli oleh setiap pelanggan.

4. ETL 1

Mengambil inputan dari file csv:

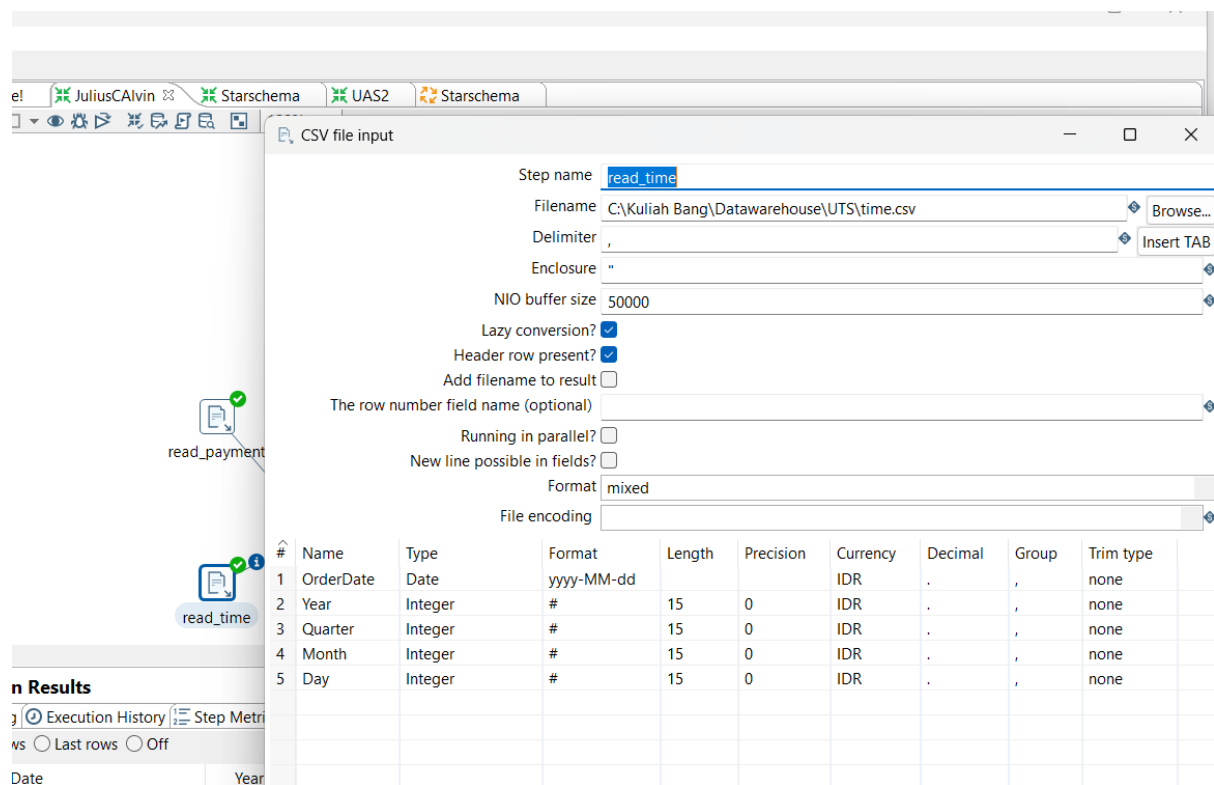
The screenshot shows the 'CSV file input' configuration window in the StarSchema UAS2 interface. The step name is 'read_payment'. The file name is 'C:\Kuliah Bang\Datawarehouse\UTS\payment.csv'. The delimiter is set to a comma (,). The enclosure is set to double quotes ("). The NIO buffer size is 50000. The 'Lazy conversion?' checkbox is checked. The 'Header row present?' checkbox is checked. The 'Add filename to result' checkbox is unchecked. The 'The row number field name (optional)' field is empty. The 'Running in parallel?' checkbox is unchecked. The 'New line possible in fields?' checkbox is unchecked. The 'Format' is set to 'mixed'. The 'File encoding' is set to 'UTF-8'. Below the configuration, a table displays the schema of the input file:

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim type
1	PaymentID	Integer	#	15	0	IDR	.	,	none
2	PaymentMethod	String		11		IDR	.	,	none
3	CreditCardType	String		10		IDR	.	,	none
4	PaymentStatus	String		4		IDR	.	,	none
5	OrderDate	Date	yyyy-MM-dd			IDR	.	,	none

Below the table, the 'Execution Results' section shows the first rows of the data:

PaymentID	PaymentMethod	CreditCardType
1	Credit Card	Visa
2	PayPal	MasterCard
3	Credit Card	Amex

Mengambil inputan dari file csv:



Step name: **read_time**

Filename: C:\Kuliah Bang\Datawarehouse\UTS\time.csv

Delimiter: ,

Enclosure: "

NIO buffer size: 50000

Lazy conversion? ☒

Header row present? ☒

Add filename to result ☐

The row number field name (optional):

Running in parallel? ☐

New line possible in fields? ☐

Format: mixed

File encoding:

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim type
1	OrderDate	Date	yyyy-MM-dd			IDR	.	,	none
2	Year	Integer	#	15	0	IDR	.	,	none
3	Quarter	Integer	#	15	0	IDR	.	,	none
4	Month	Integer	#	15	0	IDR	.	,	none
5	Day	Integer	#	15	0	IDR	.	,	none

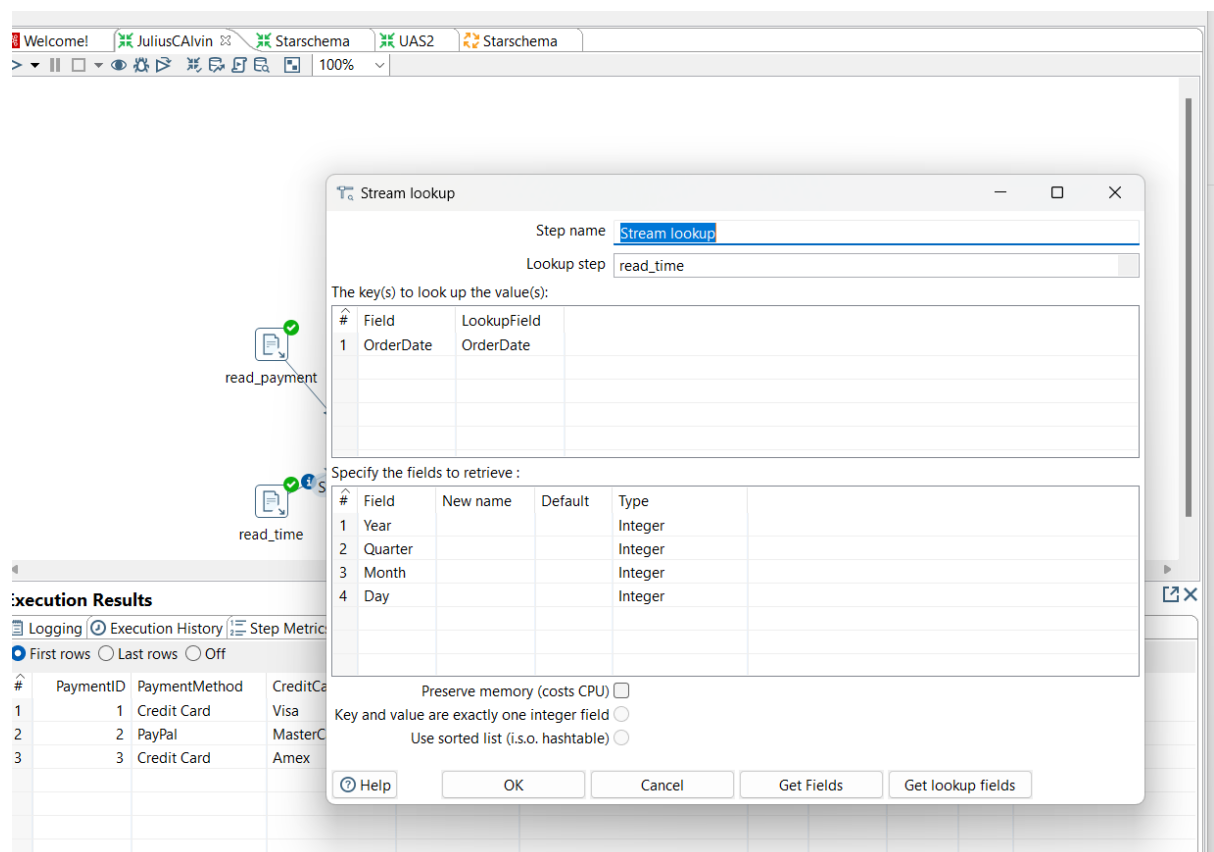
Execution Results

Logging ☒ Execution History ☐ Step Metrics ☐

First rows ☒ Last rows ☐ Off ☐

#	Date	Year
---	------	------

Stream lookup mengambil OrderDate:



Step name: **Stream lookup**

Lookup step: read_time

The key(s) to look up the value(s):

#	Field	LookupField
1	OrderDate	OrderDate

Specify the fields to retrieve:

#	Field	New name	Default	Type
1	Year			Integer
2	Quarter			Integer
3	Month			Integer
4	Day			Integer

Preserve memory (costs CPU) ☐

Key and value are exactly one integer field ☐

Use sorted list (i.s.o. hashtable) ☐

Execution Results

Logging ☒ Execution History ☐ Step Metrics ☐

First rows ☒ Last rows ☐ Off ☐

#	PaymentID	PaymentMethod	CreditCard
1	1	Credit Card	Visa
2	2	PayPal	MasterCard
3	3	Credit Card	Amex

Mengambil inputan dari file csv :

Step name: **Select values**

Select & Alter | Remove | Meta-data

Fields to alter the meta-data for :

#	Fieldname	Rename to	Type	Length	Precision
1	PaymentID		Integer	15	0
2	PaymentMethod		String	11	
3	CreditCardType		String	10	
4	PaymentStatus		String	4	
5	OrderDate		Date		
6	Year		Integer	15	0
7	Quarter		Integer	15	0
8	Month		Integer	15	0
9	Day		Integer	15	0

Get fields to change

Help OK Cancel

Preview data

PaymentStatus	OrderDate	Year	Quarter	Month	Day
	Sun Jan 15 00:00:00 WIB 2023	2023	1	1	15
	Mon Feb 20 00:00:00 WIB 2023	2023	1	2	20
	Fri Mar 10 00:00:00 WIB 2023	2023	1	3	10

Mengeluarkan output table pada database:

Step name: **Table output**

Connection: diecaast Edit... New... Wizard...

Target schema: Browse...

Target table: paymentinfo Browse...

Commit size: 1000

Truncate table ☒

Ignore insert errors ☐

Specify database fields ☒

Main options Database fields

Partition data over tables ☐

Partitioning field:

Partition data per month ☒

Partition data per day ☐

Use batch update for inserts ☒

Is the name of the table defined in a field? ☐

Field that contains name of table:

Store the tablename field ☒

Return auto-generated key ☐

Name of auto-generated key field:

Help OK Cancel SQL

Execution Results

Logging Execution History Step Metrics

First rows Last rows Off

#	PaymentID	PaymentMethod	CreditCardType
1	1	Credit Card	Visa
2	2	PayPal	MasterCard
3	3	Credit Card	Amex

Hasilnya adalah:

The diagram shows an ETL pipeline with the following steps: CSV file input, inputamount, Stream lookup, Select values, and Table output. Below the diagram is a screenshot of the 'Execution Results' window, which displays a table of payment data.

Execution Results

Logging: ☒ Execution History ☐ Step Metrics ☐ Performance Graph ☐ Metrics ☒ Preview data

First rows: ☐ Last rows ☐ Off

PaymentID	PaymentMethod	CreditCardType	PaymentStatus	OrderDate	Year	Quarter	Month	Day	AmountID	TotalAmo
1	Credit Card	Visa	Paid	Sun Jan 15 00:00:00 WIB 2023	2023	1	1	15	1	29
2	PayPal	MasterCard	Paid	Mon Feb 20 00:00:00 WIB 2023	2023	1	2	20	2	19
3	Credit Card	Amex	Paid	Fri Mar 10 00:00:00 WIB 2023	2023	1	3	10	3	9

5. ETL 2

Mengambil inputan dari file csv:

The screenshot shows the 'CSV file input' configuration window. The 'Step name' is 'S'. The 'Filename' is 'C:\Kuliah Bang\Datawarehouse\UTS\paymentinfo.csv'. The 'Delimiter' is ','. The 'Enclosure' is '"'. The 'NIO buffer size' is '50000'. The 'Lazy conversion?' checkbox is checked. The 'Header row present?' checkbox is checked. The 'Add filename to result' checkbox is unchecked. The 'The row number field name (optional)' field is empty. The 'Running in parallel?' checkbox is unchecked. The 'New line possible in fields?' checkbox is unchecked. The 'Format' is 'mixed'. The 'File encoding' is empty. Below the configuration fields is a table showing the data types and formats for the CSV file.

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Tri
1	PaymentID	Integer	#	15	0	IDR	.	,	no
2	PaymentMethod	String		11		IDR	.	,	no
3	CreditCardType	String		10		IDR	.	,	no
4	PaymentStatus	String		4		IDR	.	,	no
5	OrderDate	Date	yyyy-MM-dd HH:mm:ss			IDR	.	,	no
6	Year	Integer	#	15	0	IDR	.	,	no
7	Quarter	Integer	#	15	0	IDR	.	,	no
8	Month	Integer	#	15	0	IDR	.	,	no
9	Day	Integer	#	15	0	IDR	.	,	no

Mengambil inputan dari file csv:

CSV file input

Step name inputamount

Filename C:\Kuliaah Bang\Datawarehouse\UTS\amount.csv Browse...

Delimiter ,

Enclosure "

NIO buffer size 50000

Lazy conversion? ☒

Header row present? ☒

Add filename to result ☐

The row number field name (optional)

Running in parallel? ☐

New line possible in fields? ☐

Format mixed

File encoding

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim type
1	AmountID	Integer	#	15	0	IDR	.	,	none
2	PaymentID	Integer	#	15	0	IDR	.	,	none
3	TotalAmount	Number	#,.	5	2	IDR	.	,	none
4	QuantitySold	Integer	#	15	0	IDR	.	,	none

? Help OK Get Fields Preview Cancel

Melakukan Stream lookup untuk mengambil AmountID, TotalAmount dan QuantitySold

Stream lookup

— □ ×

Step name

Stream lookup

Lookup step

inputamount

The key(s) to look up the value(s):

#	Field	LookupField
1	PaymentID	PaymentID

Specify the fields to retrieve :

#	Field	New name	Default	Type
1	AmountID			Integer
2	TotalAmount			Number
3	QuantitySold			Integer

Preserve memory (costs CPU) ☒

Key and value are exactly one integer field ☐

Use sorted list (i.s.o. hashtable) ☐

Help

OK

Cancel

Get Fields

Get lookup fields

Melakukan select Values

Select values

Step name

Select & Alter Remove **Meta-data**

Fields to alter the meta-data for :

#	Fieldname	Rename to	Type	Length	Precision
1	PaymentID		None	15	0
2	PaymentMethod		None	11	
3	CreditCardType		None	10	
4	PaymentStatus		None	4	
5	OrderDate		None		
6	Year		None	15	0
7	Quarter		None	15	0
8	Month		None	15	0
9	Day		None	15	0
1..	AmountID		None	15	0
1..	TotalAmount		None	5	2
1..	QuantitySold		None	15	0

Get fields to change

Help OK Cancel

Melakukan table output pada database Bernama "Paymentinfo:

Table output

Step name: Table output

Connection: diecast

Target schema: diecastnew

Target table: paymentinfo

Commit size: 1000

Truncate table: ☒

Ignore insert errors: ☐

Specify database fields: ☒

Main options

Database fields

Partition data over tables: ☐

Partitioning field:

Partition data per month: ☒

Partition data per day: ☐

Use batch update for inserts: ☒

Is the name of the table defined in a field?: ☐

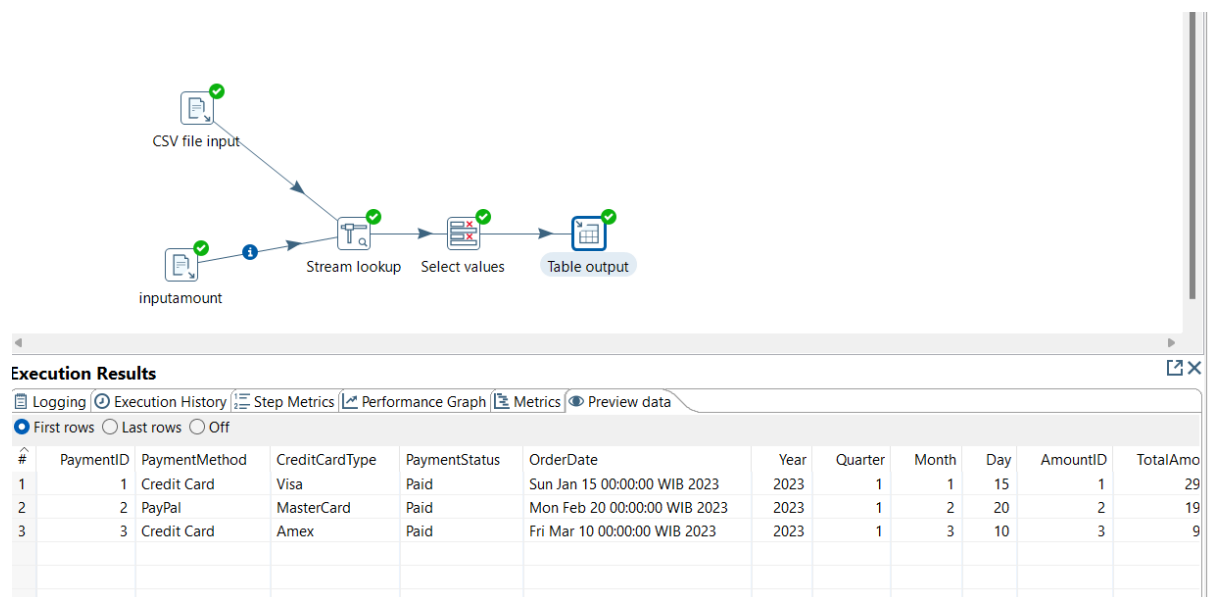
Field that contains name of table:

Store the tablename field: ☒

Return auto-generated key: ☐

Name of auto-generated key field:

Help OK Cancel SQL



6. Job (KJB)

Pada JOB ini akan melakukan ETL1, ETL2 dan starschema saat di run.

Melakukan pengimplementasi ETL1 pada transformasi yang pertama

The screenshot displays a job flow diagram at the top and a detailed configuration window for the 'ETL1' transformation below.

Job Flow Diagram:

- A 'Start' node (green play button icon) branches into two parallel paths.
- The top path contains an 'ETL1' transformation node (green square with four arrows) followed by a 'Starschema' transformation node (green square with four arrows).
- The bottom path contains an 'ETL2' transformation node (green square with four arrows).
- Both paths converge into a single green arrow pointing to the 'Starschema' node.
- Orange padlock icons are visible on the arrows connecting 'Start' to 'ETL1' and 'Start' to 'ETL2'.

Transformation Configuration Window (ETL1):

- Entry Name:** ETL1
- Transformation:** C:\Kuliah Bang\Datawarehouse\UTS\ETL1.ktr (with a 'Browse...' button)
- Options Tab:**
 - Run configuration:** (empty dropdown menu)
 - Execution:**
 - ☐ Execute every input row
 - ☐ Clear results rows before execution
 - ☐ Clear results files before execution
 - ☒ Wait for remote transformation to complete
 - ☐ Follow local abort to remote transformation
 - ☐ Suppress result data from remote transformation

Melakukan pengimplementasi ETL2 pada transformasi yang kedua

Calvin Starschema UAS2 Starschema

100%

Start ETL1 ETL2 Starschema

Transformation

Entry Name: ETL2

Transformation: C:/Kuliah Bang/Datawarehouse/UTS/UAS2.ktr Browse...

Options Logging Arguments Parameters

Run configuration:

Execution

- ☐ Execute every input row
- ☐ Clear results rows before execution
- ☐ Clear results files before execution
- ☒ Wait for remote transformation to complete
- ☐ Follow local abort to remote transformation
- ☐ Suppress result data from remote transformation

Melakukan pengimplementasi starschema pada transformasi yang ketiga

The screenshot displays a data transformation tool interface. At the top, there are several tabs: 'Welcome!', 'JuliusCALvin', 'Starschema', 'UAS2', and 'Starschema'. Below the tabs, a workflow diagram is visible, showing a 'Start' node branching into two paths: one leading to 'ETL1' and another to 'ETL2'. Both 'ETL1' and 'ETL2' nodes have a green checkmark, indicating successful execution. A green arrow points from 'ETL1' to a 'Starschema' node, which also has a green checkmark. The 'Starschema' node is the final destination in the workflow.

In the foreground, a 'Transformation' dialog box is open. The 'Entry Name' field is set to 'Starschema'. The 'Transformation' field shows the path 'C:\Kuliah Bang\Datawarehouse\UTS\Starschema.k' with a 'Browse...' button. The 'Options' tab is selected, showing the 'Run configuration' section. Under 'Execution', the following options are listed:

- ☐ Execute every input row
- ☐ Clear results rows before execution
- ☐ Clear results files before execution
- ☒ Wait for remote transformation to complete
- ☐ Follow local abort to remote transformation
- ☐ Suppress result data from remote transformation