

## Tugas 4

Dalam tugas ini, diberikan lima dataset yang berisi tentang data penjualan suatu perusahaan *e-commerce* bernama PT Ecomindo Digital. Kelima dataset tersebut akan digunakan untuk pengolahan data, oleh karena itu, sebelum diolah, data dibersihkan terlebih dahulu sebelum digunakan. Kelima dataset antara lain:

- *Customers*: berisi data diri customer, seperti nama, email, kota tempat tinggal, dan tanggal gabung
- *Products*: berisi detail produk yang dijual seperti nama produk, kategori, dan harga
- *Orders*: berisi detail pemesanan, seperti siapa customer yang memesan, kapan pemesanan dilakukan, dan status pemesanannya
- *Order items*: berisi detail jumlah barang yang dipesan dan apa produk yang dipesan
- *Payment*: berisi detail pembayaran, seperti metode pembayaran, tanggal pembayaran, dan jumlah yang dibayarkan.

Pembersihan data dilakukan untuk dapat menjawab pertanyaan bisnis berikut:

1. Siapa pelanggan yang paling loyal berdasarkan jumlah order dan total nilai belanja?
2. Produk apa yang paling sering dibeli ulang oleh pelanggan?
3. Berapa rata-rata waktu keterlambatan pembayaran pelanggan?
4. Bagaimana segmentasi pelanggan berdasarkan total belanja (Platinum, Gold, Silver)?

Namun, sebelum memasuki tahap pembersihan data, diidentifikasi terlebih dahulu mengenai anomali apa saja yang ada dalam setiap dataset. Setelah itu, ditentukan data apa saja yang diambil untuk menjawab keempat pertanyaan tersebut.

### I. Identifikasi Anomali

Berikut ini adalah temuan data anomali yang terdapat dalam lima dataset:

1. Dataset *Customers*: dalam dataset ini, terdiri dari lima kolom, yaitu id\_customer, nama, email, city, tanggal\_gabung. Anomali yang ditemukan terletak pada kolom nama, email, kota, dan tanggal\_gabung. Pada kolom nama terdapat customer yang mencantumkan gelar dan ada beberapa customer tidak mencantumkan gelar. Pada email terlihat bahwa terdapat banyak email yang kosong dan terdapat satu sel yang terisi dengan “@@”.

id_customer	nama	email	city	tanggal_gabung
8	Ir. Rahmat Yolanda, S.IP	Mojokert	7/4/2023 0:00	
74	Panca Manuillang	Mataram	12/17/2022 0:00	
80	Wariji Megantara	Bukitting	5/17/2024 0:00	
102	Bakiman Hartati	Banjarm	1/5/2023 0:00	
105	Bakiman Mardhiyah	Tomohor	11/23/2022 0:00	
108	R.A. Eva Samosir, S.T.	Pasuruar	8/18/2023 0:00	
169	Adhilarja Napitupulu, S.Gz	Gorontal	8/18/2024 0:00	
210	Wira Widiasiuti	Pekalong	1/18/2025 0:00	
237	Tgk. Prasetyo Sitorus	Payakum	10/20/2022 0:00	
279	Carla Palastri, S.Ked	Lhokseur	7/13/2022 0:00	
296	Wakiman Haryanti	Sabang	4/15/2025 0:00	
306	H. Jinawi Mandala		7/4/2024 0:00	
335	R. Luis Nababan		1/15/2024 0:00	
362	Nadia Gunarto	Kupang	8/7/2022 0:00	
405	Ajeng Safitri	Pasuruar	9/21/2022 0:00	
412	Prakosa Santoso	Bandung	2/22/2023 0:00	
447	Gilda Nugroho	Pasuruar	8/12/2023 0:00	
448	Darimin Nashiruddin	Pematani	7/14/2024 0:00	
449	Putri Prasetyo	Tangerar	4/3/2025 0:00	
452	Kamila Suwarno	Sungai Pi	1/20/2025 0:00	
461	Cawuk Puspasari	Cilegon	4/1/2024 0:00	
465	Melinda Kurniawan, S.Pd	Meulabu	6/13/2024 0:00	
480	Yani Haryanto	@@	Lubuklini	7/30/2023 0:00
489	Baldin Megantara, M.TI.	Prabumu	11/25/2023 0:00	
493	Maria Prasasta	Bukitting	2/18/2025 0:00	

Pada kolom kota juga terdapat beberapa yang kosong. Pada kolom tanggal\_gabung, terdapat beberapa sel yang tanggalnya aneh dan formatnya tidak sesuai (harusnya tertulis dalam format MM/dd/yyyy namun malah tertulis dalam format dd-MM-yyyy)

<b>id_customer</b>	<b>nama</b>	<b>email</b>	<b>city</b>	<b>tanggal_gabung</b>
132	Viktor Wijaya	viktorwiji	Sibolga	31-31-1400
152	Mumpuni Zulaika	mumpun	Padang	45-12-1501
155	Rahayu Utami, S.Ked	rahayu.u	Meulaboh	99-31-1800
191	Cut Siti Marbun	cut.siti60	Palopo	05-44-2105
425	Puput Riyanti	puput_ri	Bandung	33-15-2020

2. *Products*: dalam dataset ini, terdapat empat kolom, yaitu id\_product, nama\_produk, kategori, dan harga. Anomali ditemukan hanya pada kolom harga, yaitu terdapat produk yang memiliki penulisan harganya kurang tepat (ada simbol "-") dan terdapat 2 produk yang harganya tidak tercantum.

<b>id_product</b>	<b>nama_produk</b>	<b>kategori</b>	<b>harga</b>
1	Sabun Mandi	Kebutuhan	-8000
11	Teh Celup	Minuman	
15	Susu UHT	Minuman	
18	Sikat Gigi	Kebutuhan	-4500

3. *Orders*: dalam dataset ini, terdapat empat kolom, yaitu id\_order, id\_customer, tanggal\_order, dan status\_order. Terdapat anomali yang ditemukan dalam kolom id\_customer, tanggal\_order, dan status\_order. Pertama, dalam id\_customer terdapat id\_customer 501-509, padahal jika mengacu pada dataset Customer, id\_customer hanya sampai 500. Kedua, dalam tanggal\_order, terdapat format tanggal yang tidak terbaca sebagai tanggal meskipun penulisannya sudah benar dan terdapat tanggal yang kosong. Ketiga, pada status\_order, terdapat status order yang tidak terisi (kosong), TRUE, dan FALSE. Ketiga anomali tersebut dapat dilihat pada gambar di bawah ini.

<b>id_customer</b>	<b>status_order</b>
506	FALSE
509	FALSE
508	TRUE
505	
501	FALSE
509	TRUE
509	TRUE
506	
502	FALSE
507	TRUE

4. *Order\_items*: dalam dataset order\_items, terdapat empat kolom, yaitu id\_order\_item, id\_order, id\_product, dan quantity. Terdapat tiga anomali yang ditemukan dalam kolom quantity. Pertama, terdapat jumlah order yang kosong (tidak ada data dalam kolom quantity). Kedua, quantity yang dipesan tidak bulat (contoh: 0.5, 0.6, 2.1, dll). Ketiga, terdapat inputasi yang tidak cocok (menggunakan string seperti empat botol, satu pcs, dll.). Ketiga anomali dapat dilihat pada gambar di bawah ini.

quantity
2.1
0.5
0.5
0.9
0.6
0.5

quantity
satu pcs
tiga
lima unit
lima unit
lima unit
empat botol
empat botol

5. *Payments*: dalam dataset ini, terdapat lima kolom, antara lain id\_payment, id\_order, tanggal\_bayar, metode\_bayar, dan amount\_paid. Dalam dataset ini tidak ditemukan anomali apapun.

## II. Persiapan dalam MySQL

Sebelum melakukan pembersihan data dalam Talend, data akan di load terlebih dahulu di MySQL. Hal ini bertujuan untuk menyaring kolom mana saja yang akan diambil untuk pengolahan data, karena tidak semua kolom yang ada dalam masing-masing dataset akan digunakan. Untuk melakukan loading ke dalam MySQL, dibuat lima tabel dengan masing-masing nama seperti dengan nama dataset. Berikut ini adalah pengaturan yang digunakan ketika membuat kelima tabel

```

CREATE TABLE customers (
    id_customer INT,
    nama VARCHAR(255),
    email VARCHAR(255),
    city VARCHAR(255),
    tanggal_gabung VARCHAR(255),
    PRIMARY KEY (id_customer)
);

CREATE TABLE products (
    id_product INT,
    nama_produk VARCHAR(255),
    kategori VARCHAR(255),
    harga VARCHAR(255),
    PRIMARY KEY (id_product)
);

CREATE TABLE orders (
    id_order INT,
    id_customer INT,
    tanggal_order VARCHAR(255),
    status_order VARCHAR(255),
    PRIMARY KEY (id_order),
    FOREIGN KEY (id_customer) REFERENCES customers(id_customer)
);

CREATE TABLE order_items (
    id_order_item INT,
    id_order INT,
    id_product INT,
    quantity VARCHAR(255),
    PRIMARY KEY (id_order_item),
    FOREIGN KEY (id_order) REFERENCES orders(id_order),
    FOREIGN KEY (id_product) REFERENCES products(id_product)
);

CREATE TABLE payments (
    id_payment INT,
    id_order INT,
    tanggal_bayar DATE,
    metode_bayar VARCHAR(255),
    amount_paid INT,
    PRIMARY KEY (id_payment),
    FOREIGN KEY (id_order) REFERENCES orders(id_order)
);

```

Perhatikan bahwa untuk beberapa kolom yang harusnya berformat tanggal (DATE) atau numerik (INT), diatur menjadi string terlebih dahulu. Hal ini dilakukan agar seluruh dataset bisa ter-import. Misalkan jika pada tabel customers, kolom tanggal\_gabung sudah diatur menjadi DATE maka tanggal-tanggal yang tidak terkualifikasi sebagai DATE akan langsung di-drop secara otomatis oleh MySQL, yang artinya satu baris penuh akan dihapus. Hal ini tidak diinginkan, karena informasi lain seperti ID, nama, email, tetap dibutuhkan. Kolom harga yang terdapat dalam tabel products juga diatur agar menjadi string terlebih dahulu, agar baris yang tidak memiliki harga tidak di-drop oleh MySQL. Semua tipe data yang sekiranya akan membuat

MySQL membuang baris tertentu jika ada kolom yang kosong atau format tidak sesuai, diatur menjadi varchar terlebih dahulu. Tentu tipe-tipe data yang tidak sesuai nantinya akan disesuaikan lagi dalam Talend.

Selanjutnya, dibuat empat *view* yang akan berisi kolom yang berguna untuk menjawab keempat pertanyaan yang sudah disebutkan dalam pendahuluan.

1. *View* 1: *view\_customer\_loyalty*. Dalam *view* ini, harapannya *data analyst* dapat menjawab pertanyaan “siapa pelanggan yang paling loyal berdasarkan jumlah order dan total nilai belanja?”. Oleh karena itu, setidaknya dalam *view* ini memuat kolom-kolom seperti id customer, nama, email (sebagai informasi pelengkap), jumlah barang yang sudah dipesan, dan total bayarnya. Kode yang digunakan dapat dilihat pada gambar di bawah ini

```
CREATE VIEW view_customer_loyalty AS
SELECT
    c.id_customer,
    c.nama,
    c.email,
    oi.quantity,
    o.status_order,
    p.amount_paid
FROM
    customers c
LEFT JOIN orders o ON c.id_customer = o.id_customer
LEFT JOIN order_items oi ON o.id_order = oi.id_order
LEFT JOIN payments p ON o.id_order = p.id_order;
```

Tabel yang dihasilkan:

id_customer	nama	email	quantity	status_order	amount_paid
1	Reza Namaga	reza.namaga539@yahoo.com	5.0	completed	49000
1	Reza Namaga	reza.namaga539@yahoo.com	5.0	completed	49000
1	Reza Namaga	reza.namaga539@yahoo.com	2.0	cancelled	NULL
2	Dr. Ani Simbolon	dr.anis257@yahoo.co.id	1.0	cancelled	NULL
2	Dr. Ani Simbolon	dr.anis257@yahoo.co.id	5.0	cancelled	NULL
2	Dr. Ani Simbolon	dr.anis257@yahoo.co.id	2.0	cancelled	NULL
3	Ir. Raden Sirombing, S.Farm	ir_raden510@gmail.co.id	NULL	NULL	NULL
4	R. Ami Damanik	r.ami717@outlook.com	3.0	completed	94700
4	R. Ami Damanik	r.ami717@outlook.com	4.0	completed	94700
4	R. Ami Damanik	r.ami717@outlook.com	5.0	completed	94700

Perhatikan bahwa dari tabel yang dihasilkan, id customer 1 terlihat seperti berbelanja sebanyak 98000 (49000+49000), namun sebenarnya ia hanya berbelanja dengan total 49000 untuk total 10 barang. Hal ini dapat dilihat pada tabel *orders*, *order\_items*, dan *payments*. Pertama dari tabel *payments*, dapat dilihat bahwa untuk id\_order 212, terjadi pembayaran sebesar 49000

id_payment	id_order	tanggal_bayar	metode_bayar	amount_paid
112	212	4/7/2024	E-Wallet	49000

Lalu, dengan mencocokan id\_order yang ada di tabel *order\_items*, dapat diidentifikasi bahwa pembayaran 49000 tersebut untuk 10 barang.

id_order_item	id_order	id_product	quantity
432	212	16	5
433	212	11	5

Terakhir, dengan mencocokan id\_order yang ada di table *orders*, maka dapat diketahui bahwa *customer* yang memesan barang dengan id order 212 adalah milik id customer 1

<b>id_order</b>	<b>id_customer</b>	<b>tanggal_order</b>	<b>status_order</b>
212	1	4/4/2024	completed
899	1	9/25/2023	cancelled

Jadi dapat disimpulkan bahwa id customer 1 melakukan pembayaran sebesar 49000 untuk 10 barang, bukan 49000 untuk masing-masing 5 barang. Untuk mengubah hal ini, akan dilakukan di Talend.

- View 2: view\_product\_repeatability. Dalam view ini, harapannya *data analyst* dapat menjawab pertanyaan “Produk apa yang paling sering dibeli ulang oleh pelanggan?”. Oleh karena itu, dibutuhkan informasi berupa nama produk, jumlah barang yang dipesan, dan kapan produk tersebut dipesan. Kode yang digunakan untuk membuat view dapat dilihat pada gambar di bawah ini

```
CREATE VIEW view_product_repeatability AS
SELECT
    oi.id_order_item,
    p.id_product,
    p.nama_produk,
    p.kategori,
    o.status_order,
    oi.quantity,
    o.tanggal_order
FROM
    order_items oi
LEFT JOIN products p ON oi.id_product = p.id_product
LEFT JOIN orders o ON oi.id_order = o.id_order;
```

Tabel yang dihasilkan:

<b>id_order_item</b>	<b>id_product</b>	<b>nama_produk</b>	<b>kategori</b>	<b>status_order</b>	<b>quantity</b>	<b>tanggal_order</b>
1	10	Saus Sambal	Makanan	cancelled	4.0	2024-11-20
2	16	Sabun Cuci Piring	Kebutuhan Rumah Tangga	cancelled	5.0	2023-05-23
3	14	Air Mineral 600ml	Minuman	cancelled	4.0	2023-05-23
4	19	Gula Pasir 1Kg	Makanan	cancelled	1.0	2023-05-23
5	11	Teh Celup	Minuman	completed	1.0	2025-02-17
6	15	Susu UHT	Minuman	cancelled	5.0	2025-01-16

- View 3: view\_payment\_behavior. Dalam view ini harus berisi data-data yang dapat menjawab pertanyaan “Berapa rata-rata waktu keterlambatan pembayaran pelanggan?”. Oleh karena itu, setidaknya harus memuat informasi berupa tanggal pemesanan dan tanggal bayar. Kode yang digunakan untuk membuat view ini yaitu

```
CREATE VIEW view_payment_behavior AS
SELECT
    pa.id_payment,
    pa.tanggal_bayar,
    pa.metode_bayar,
    pa.amount_paid,
    o.tanggal_order,
    o.status_order
FROM
    payments pa
LEFT JOIN orders o ON pa.id_order = o.id_order;
```

Tabel yang dihasilkan:

id_payment	tanggal_bayar	metode_bayar	amount_paid	tanggal_order	status_order
2	2023-07-27	Kartu Kredit	83200	2023-07-26	completed
3	2025-03-04	Transfer Bank	27200	2025-03-02	completed
4	2024-12-12	E-Wallet	40600	2024-12-12	completed
5	2024-09-29	E-Wallet	47500	2024-09-28	completed
6	2024-08-02	Kartu Kredit	20500	2024-07-31	completed
7	2023-08-12	E-Wallet	33400	2023-08-10	completed

4. View 4: view\_customer\_segmentation. Tabel yang ada dalam view ini harus memuat informasi yang cukup untuk menjawab pertanyaan “Bagaimana segmentasi pelanggan berdasarkan total belanja (Platinum, Gold, Silver)?”. Oleh karena itu, dalam view ini setidaknya memuat informasi mengenai nam customer, total uang yang sudah dibayarkan, dan segmentasi dari pelanggan. Kode yang digunakan yaitu

```
CREATE VIEW view_customer_segmentation AS
SELECT
    c.id_customer,
    c.nama,
    COALESCE(total_spending.total_belanja, 0) AS total_belanja,
    CASE
        WHEN COALESCE(total_spending.total_belanja, 0) > 10000000 THEN 'Platinum'
        WHEN COALESCE(total_spending.total_belanja, 0) BETWEEN 5000000 AND 10000000 THEN 'Gold'
        ELSE 'Silver'
    END AS kategori_pelanggan
FROM
    customers c
LEFT JOIN (
    SELECT
        o.id_customer,
        SUM(CAST(p.amount_paid AS UNSIGNED)) AS total_belanja
    FROM
        payments p
    JOIN orders o ON p.id_order = o.id_order
    GROUP BY o.id_customer
) AS total_spending ON c.id_customer = total_spending.id_customer;
```

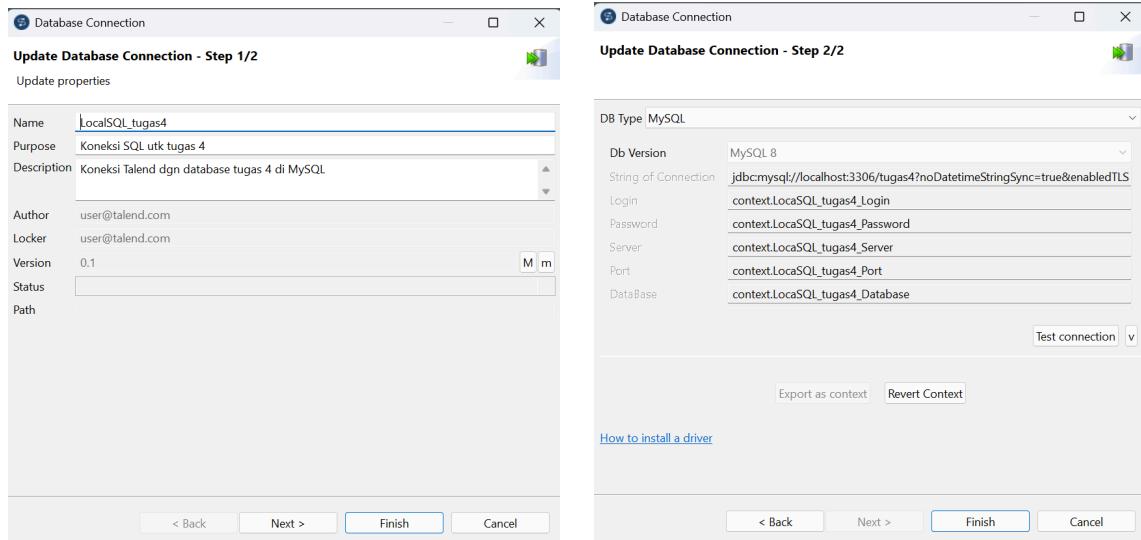
Tabel yang dihasilkan:

id_customer	nama	total_belanja	kategori_pelanggan
1	Reza Namaga	49000	Silver
2	Dr. Ani Simbolon	0	Silver
3	Ir. Raden Sihombing, S.Farm	0	Silver
4	R. Ami Damanik	184900	Silver
5	Mitra Namaga, M.TI.	0	Silver
6	Restu Situmorang	0	Silver

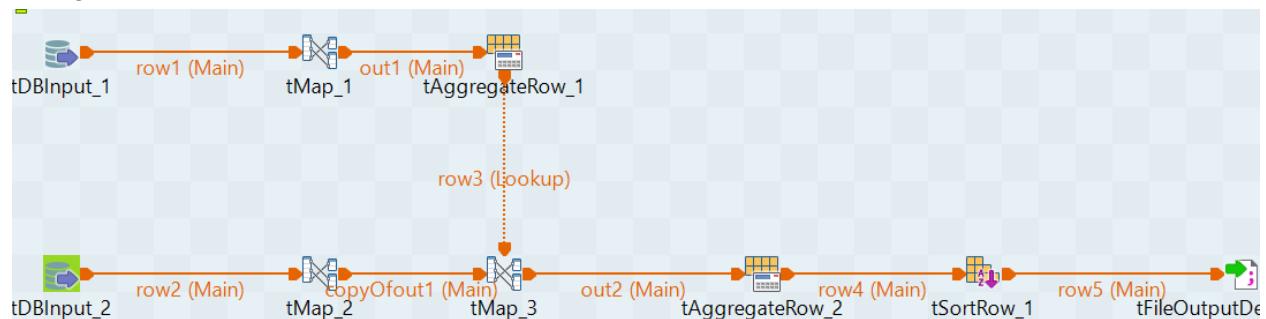
Setelah sudah membuat empat view tersebut maka langkah selanjutnya adalah dengan melakukan proses load dan transform di dalam Talend.

### III. Proses dalam Talend

Untuk menghubungkan antara Talend dan MySQL, dibuat sebuah koneksi dengan Db Connections dalam Metadata. Detail dari koneksi yang dibuat dapat dilihat pada gambar di bawah.



Setelah membuat koneksi, pada job telah dibuat, penulis akan mentransformasi keempat view sesuai dengan urutan. Untuk view\_customer\_loyalty, dibuat alur transformasi sebagai berikut



Perlu diketahui sebelumnya bahwa tDBInput\_2 dan tMap\_2 hanya duplikat dari tDBInput\_1 dan tMap\_1. Pada tDBInput (tMySQLInput), seluruh kolom dalam view\_customer\_loyalty dipanggil dan langsung menghubungkannya ke dalam tMap. Dalam tMap terjadi transformasi sebagai berikut:

- Pertama, pembersihan nama setiap customer. Customer yang mencantumkan gelar setelah nama lengkapnya, gelar tersebut dihapus. Hal ini dilakukan karena pada tFileOutputDelimited, pemisah yang ditetapkan adalah ",". Jika tetap mempertahankan gelar maka Talend akan memindahkan gelar ke kolom baru (karena gelar umumnya ditulis setelah nama dengan pemisah ","). Kode yang digunakan: row1.nama != null ? row1.nama.split(",")[0].trim() : "unknown".
- Kedua, mengubah email "@@" dan mengisi email kosong. Keduanya diperlakukan sama, yaitu sama-sama diisi dengan "unknown". Kode yang digunakan: (row1.email ==

```
null || row1.email.trim().isEmpty() || row1.email.trim().contains("@@@")) ? "unknown" :  
row1.email
```

- Ketiga, mengubah data yang ada di kolom quantity (seperti 0.5, 0.6, 1.7, 2 botol, satu pcs) menjadi bilangan bulat. Untuk itu, dibuat sebuah routine baru dalam Code. Routine tersebut dinamakan StrToNumber, detailnya dapat dilihat pada gambar di bawah ini

```
public static Integer toInteger(String input) {  
    if (input == null || input.trim().isEmpty()) {  
        return null;  
    }  
  
    input = input.toLowerCase().trim();  
  
    // Word-based mappings  
    if (input.contains("satu")) return 1;  
    if (input.contains("dua")) return 2;  
    if (input.contains("tiga")) return 3;  
    if (input.contains("empat")) return 4;  
    if (input.contains("lima")) return 5;  
    if (input.contains("enam")) return 6;  
    if (input.contains("tujuh")) return 7;  
    if (input.contains("delapan")) return 8;  
    if (input.contains("sembilan")) return 9;  
  
    // Extract numeric part (keep dot for decimal numbers)  
    String numeric = input.replaceAll("[^0-9.]", "");  
    if (!numeric.isEmpty()) {  
        try {  
            double value = Double.parseDouble(numeric);  
            return (int) Math.ceil(value); // Cast to Integer safely  
        } catch (NumberFormatException e) {  
            return null; // Return null on invalid format  
        }  
    }  
}
```

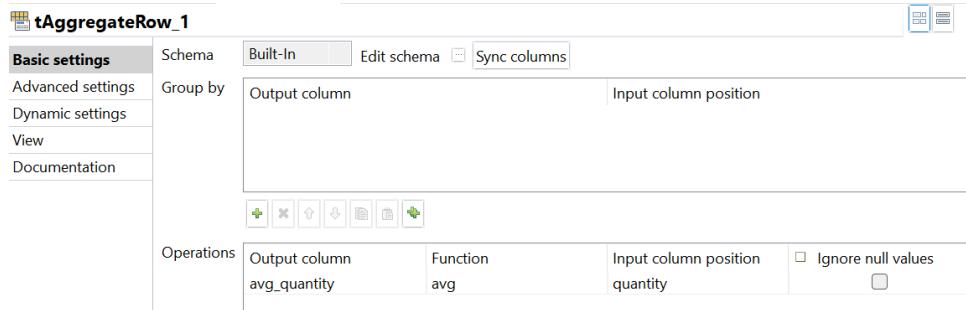
Dari gambar tersebut, selain mengubah string seperti satu pcs, empat botol menjadi 1, 5, dilakukan juga pembulatan ke atas. Jadi nilai-nilai seperti 0.5 atau 0.6 dibulatkan menjadi 1, nilai seperti 1.6 atau 1.7 dibulatkan menjadi 2. Penggunaan routine ini di dalam tMap ditulis dengan: StrToNumber.toInteger(row1.quantity). Dengan demikian, kolom quantity yang awalnya string, bisa diubah tipe datanya menjadi integer

- Keempat, menangani status\_order. Sudah dijelaskan dalam identifikasi anomali bahwa status order memiliki anomali seperti data kosong dan data terisi TRUE/FALSE. Penulis memutuskan untuk mengisi data kosong menjadi "unknown", TRUE menjadi "completed", dan FALSE menjadi "cancelled". Kode yang digunakan: (row1.status\_order == null || row1.status\_order.trim().isEmpty()) ? "unknown" : row1.status\_order.equalsIgnoreCase("TRUE") ? "completed" : row1.status\_order.equalsIgnoreCase("FALSE") ? "cancelled" : row1.status\_order.

Column	K..	Type	N.	Date Pattern (Ct..)	Length	Precision	Default	Comment
id_customer	<input type="checkbox"/>	int	<input checked="" type="checkbox"/>		11	0		
nama	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>		255	0		
email	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>		255	0		
quantity	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>		255	0		
status_order	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>		255	0		
amount_paid	<input type="checkbox"/>	Integer	<input checked="" type="checkbox"/>		11	0		

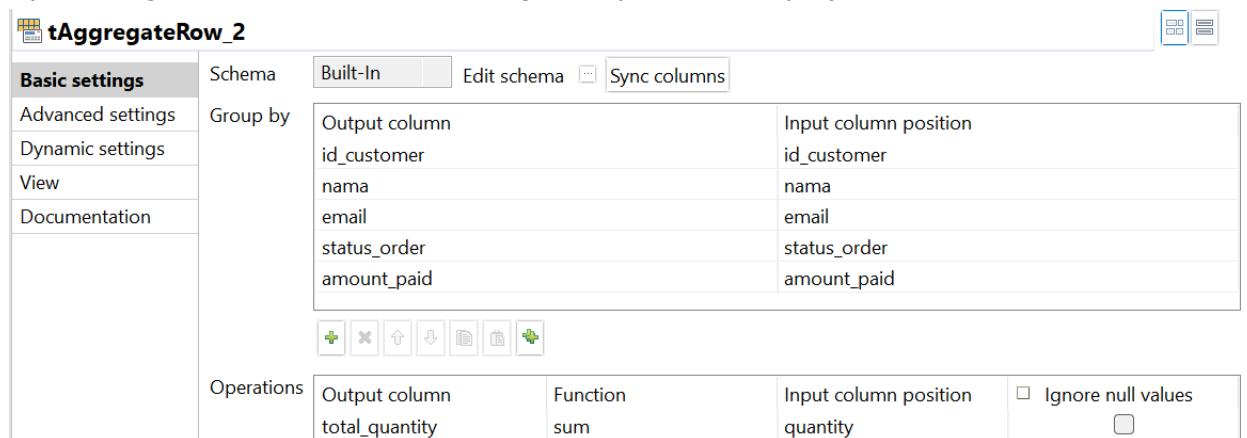
Column	K..	Type	N.	Date Pattern (Ct..)	Length	Precision	Default	Comment
id_customer	<input type="checkbox"/>	int	<input checked="" type="checkbox"/>		11	0		
nama	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>		255	0		
email	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>		255	0		
quantity	<input type="checkbox"/>	Integer	<input checked="" type="checkbox"/>		255	0		
status_order	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>		255	0		
amount_paid	<input type="checkbox"/>	Integer	<input checked="" type="checkbox"/>		11	0		

Selanjutnya, untuk alur pertama, data yang sudah ditransformasi seperti dalam tMap, di agregasi dengan tAggregateRow untuk menghitung rata-rata dari quantity. Hal ini dilakukan karena data yang kosong pada quantity belum diisi dan diputuskan untuk mengisinya dengan rata-rata.

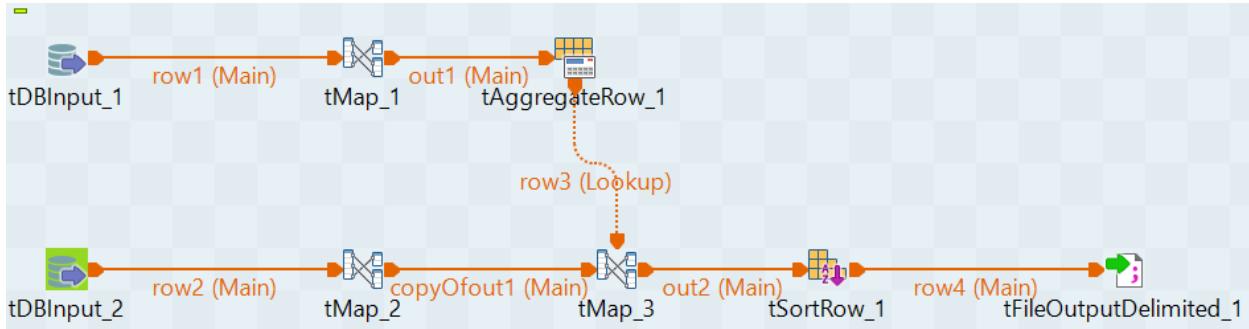


Dengan demikian, proses inputasi nilai quantity terjadi pada tMap3.

Setelah itu, untuk memecahkan masalah yang sudah dijelaskan dalam persiapan MySQL bagian view 1, dilakukan proses group by dan quantity dijumlahkan.



Setelahnya, data diurutkan berdasarkan id\_customer agar terurut dimulai dari id customer 1. Data yang sudah bersih akan disimpan ke dalam file csv bernama customer\_loyalty\_A\_6162101104. Lebih lanjut, berikut ini alur pengolahan data untuk tabel view\_product\_repeatability.



Hampir sama dengan alur pengolahan untuk view\_customer\_loyalty, tDBInput\_2 dan tMap\_2 adalah duplikat dari tDBInput\_1 dan tMap1. tMySQL (tDBInput\_1) digunakan untuk loading view\_product\_repeatability lalu langsung diproses dalam tMap. Dalam tMap proses yang dilakukan adalah:

- Mengubah TRUE/FALSE menjadi completed dan cancelled pada status\_order dan mengisi kolom kosong dengan "unknown". Kode yang digunakan: (row1.status\_order == null || row1.status\_order.trim().isEmpty()) ? "unknown" : row1.status\_order.equalsIgnoreCase("TRUE") ? "completed" : row1.status\_order.equalsIgnoreCase("FALSE") ? "cancelled" : row1.status\_order
- Mentransformasi kolom quantity seperti pada view\_customer\_loyalty. Kode yang digunakan: StrToNumber.toInteger(row1.quantity)
- Mengganti tipe data tanggal\_order (yang awalnya string) menjadi date dengan format "MM/dd/yyyy". Kode yang digunakan: row1.tanggal\_order == null || row1.tanggal\_order.trim().isEmpty() ? null : row1.tanggal\_order.matches("^\\d{4}-\\d{2}-\\d{2}\$") ? TalendDate.parseDate("yyyy-MM-dd", row1.tanggal\_order) : row1.tanggal\_order.matches("^\\d{2}/\\d{2}/\\d{4}\$") && Integer.parseInt(row1.tanggal\_order.substring(0, 2)) > 12 ? TalendDate.parseDate("dd/MM/yyyy", row1.tanggal\_order) : TalendDate.parseDate("MM/dd/yyyy", row1.tanggal\_order).

Schema editor							Expression editor										
row1							out1										
Column	K..	Type	N..	Date Pattern (Ct..)	Length	Precision	Default	Comment	Column	K..	Type	N..	Date Pattern (Ct..)	Length	Precision	Default	Comment
id_order_item	<input type="checkbox"/>	int	<input checked="" type="checkbox"/>		11	0			id_order_item	<input type="checkbox"/>	int	<input checked="" type="checkbox"/>		11	0		
id_product	<input type="checkbox"/>	Integer	<input checked="" type="checkbox"/>		11	0			id_product	<input type="checkbox"/>	int	<input checked="" type="checkbox"/>		11	0		
nama_produk	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>		255	0			nama_produk	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>		255	0		
kategori	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>		255	0			kategori	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>		255	0		
status_order	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>		255	0			status_order	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>		255	0		
quantity	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>		255	0			quantity	<input type="checkbox"/>	Integer	<input checked="" type="checkbox"/>		255	0		
tanggal_order	<input type="checkbox"/>	String	<input checked="" type="checkbox"/>		255	0			tanggal_order	<input type="checkbox"/>	Date	<input checked="" type="checkbox"/>	"MM/dd/yyyy"	255	0		

Setelah ditransformasi dalam tMap, selanjutnya diagregasi dengan tAggregateRow. Idenya hampir sama untuk menangani data kosong dalam kolom quantity, yaitu diisi dengan rata-ratanya. Oleh karena itu, rata-ratanya dihitung dalam tAggregateRow dan setelahnya baru dihubungkan di tMap\_3

tSortRow digunakan hanya untuk mengurutkan data berdasarkan id\_order\_item agar terurut dimulai dari id\_order\_item 1. Selanjutnya adalah mengolah view\_payment\_behavior. Alur untuk mengolah data view ini dapat dilihat pada gambar berikut



Jika dilihat prosesnya cukup sederhana, dari tDBInput\_1 dihubungkan ke tMap dan langsung dieksport ke file csv. Proses pembersihan yang terjadi dalam tMap adalah sebagai berikut:

- Mengubah tipe data tanggal\_order dari string menjadi date dengan format MM/dd/yyyy. Kode yang digunakan: row1.tanggal\_order == null || row1.tanggal\_order.trim().isEmpty() ? null : row1.tanggal\_order.matches("^\\d{4}-\\d{2}-\\d{2}\$") ? TalendDate.parseDate("yyyy-MM-dd", row1.tanggal\_order) : row1.tanggal\_order.matches("^\\d{2}/\\d{2}/\\d{4}\$") && Integer.parseInt(row1.tanggal\_order.substring(0, 2)) > 12 ? TalendDate.parseDate("dd/MM/yyyy", row1.tanggal\_order) : TalendDate.parseDate("MM/dd/yyyy", row1.tanggal\_order)
- Mengubah TRUE/FALSE menjadi completed dan cancelled pada status\_order dan mengisi kolom kosong dengan "unknown". Kode yang digunakan: (row1.status\_order == null || row1.status\_order.trim().isEmpty()) ? "unknown" :

- ```

row1.status_order.equalsIgnoreCase("TRUE") ? "completed" : row1.status_order.
row1.status_order.equalsIgnoreCase("FALSE") ? "cancelled" : row1.status_order.

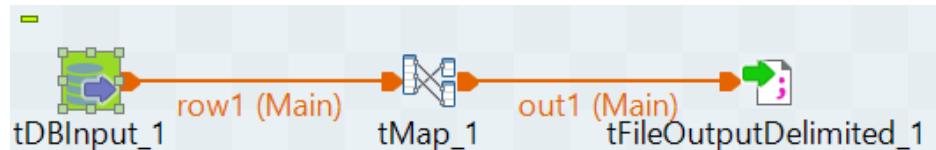
```
- Menyesuaikan format tanggal pada tanggal\_bayar dari dd-MM-yyyy menjadi MM/dd/yyyy.

| Column        | K.. Type | N. Date Pattern (Ct...) | Length | Precision | Default | Comment |
|---------------|----------|-------------------------|--------|-----------|---------|---------|
| id_payment    | int      |                         | 11     | 0         |         |         |
| tanggal_bayar | Date     | "dd-MM-yyyy"            | 10     | 0         |         |         |
| metode_bayar  | String   |                         | 255    | 0         |         |         |
| amount_paid   | Integer  |                         | 11     | 0         |         |         |
| tanggal_order | String   |                         | 255    | 0         |         |         |
| status_order  | String   |                         | 255    | 0         |         |         |

| Column        | K.. Type | N. Date Pattern (Ct...) | Length | Precision | Default | Comment |
|---------------|----------|-------------------------|--------|-----------|---------|---------|
| id_payment    | int      |                         | 11     | 0         |         |         |
| tanggal_bayar | Date     | "MM/dd/yyyy"            | 10     | 0         |         |         |
| metode_bayar  | String   |                         | 255    | 0         |         |         |
| amount_paid   | Integer  |                         | 11     | 0         |         |         |
| tanggal_order | Date     | "MM/dd/yyyy"            | 255    | 0         |         |         |
| status_order  | String   |                         | 255    | 0         |         |         |

Setelahnya baru akan disimpan dalam file csv bernama payment\_behavior\_A\_6162101104. View terakhir yang akan diolah adalah view\_customer segmentation. Sama seperti view\_payment\_behavior, alur yang digunakan untuk mengolah view\_customer\_segmentation cukup sederhana. Setelah di-load dengan tMySQL, akan langsung ditransformasi di dalam tMap dan langsung diekspor ke dalam file CSV.



Proses yang dilakukan dalam tMap hanya satu, yaitu membersihkan nama dengan cara menghapus gelar yang ada (kode yang digunakan: row1.nama != null ? row1.nama.split(",")[0].trim() : "unknown").

| Column             | K.. Type   | N. Date Pattern (Ct...) | Length | Precision | Default | Comment |
|--------------------|------------|-------------------------|--------|-----------|---------|---------|
| id_customer        | int        |                         | 11     | 0         |         |         |
| nama               | String     |                         | 255    | 0         |         |         |
| total_belanja      | BigDeci... |                         | 43     | 0         |         |         |
| kategori_pelanggan | String     |                         | 8      | 0         |         |         |

| Column             | K.. Type   | N. Date Pattern (Ct...) | Length | Precision | Default | Comment |
|--------------------|------------|-------------------------|--------|-----------|---------|---------|
| id_customer        | int        |                         | 11     | 0         |         |         |
| nama               | String     |                         | 255    | 0         |         |         |
| total_belanja      | BigDeci... |                         | 43     | 0         |         |         |
| kategori_pelanggan | String     |                         | 8      | 0         |         |         |

