

**IF2240 - BASIS DATA**  
**MILESTONE 3**  
**DESAIN BASIS DATA RELASIONAL DAN IMPLEMENTASI**  
***BUSINESS RULE***



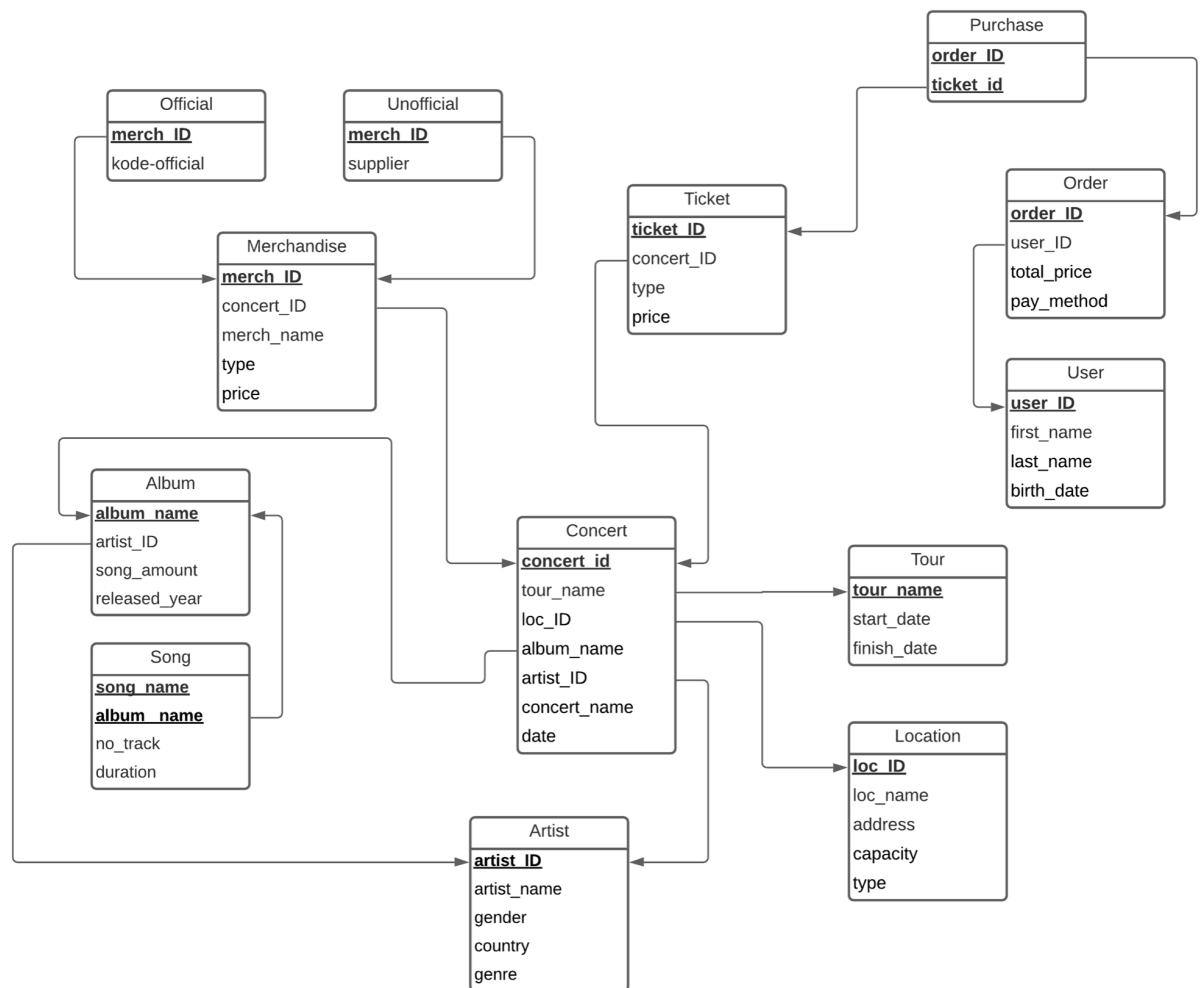
**Kelompok 07 K-04:**

**13519175 Stefanus Jeremy Aslan**  
**13519179 Akifa Nabil Ufairah**  
**13519185 Richard Rivaldo**  
**13519199 Christian Gunawan**  
**13519201 Muhammad Rayhan Ravianda**

**Program Studi Teknik Informatika**  
**Sekolah Teknik Elektro dan Informatika**  
**Institut Teknologi Bandung**  
**2021**

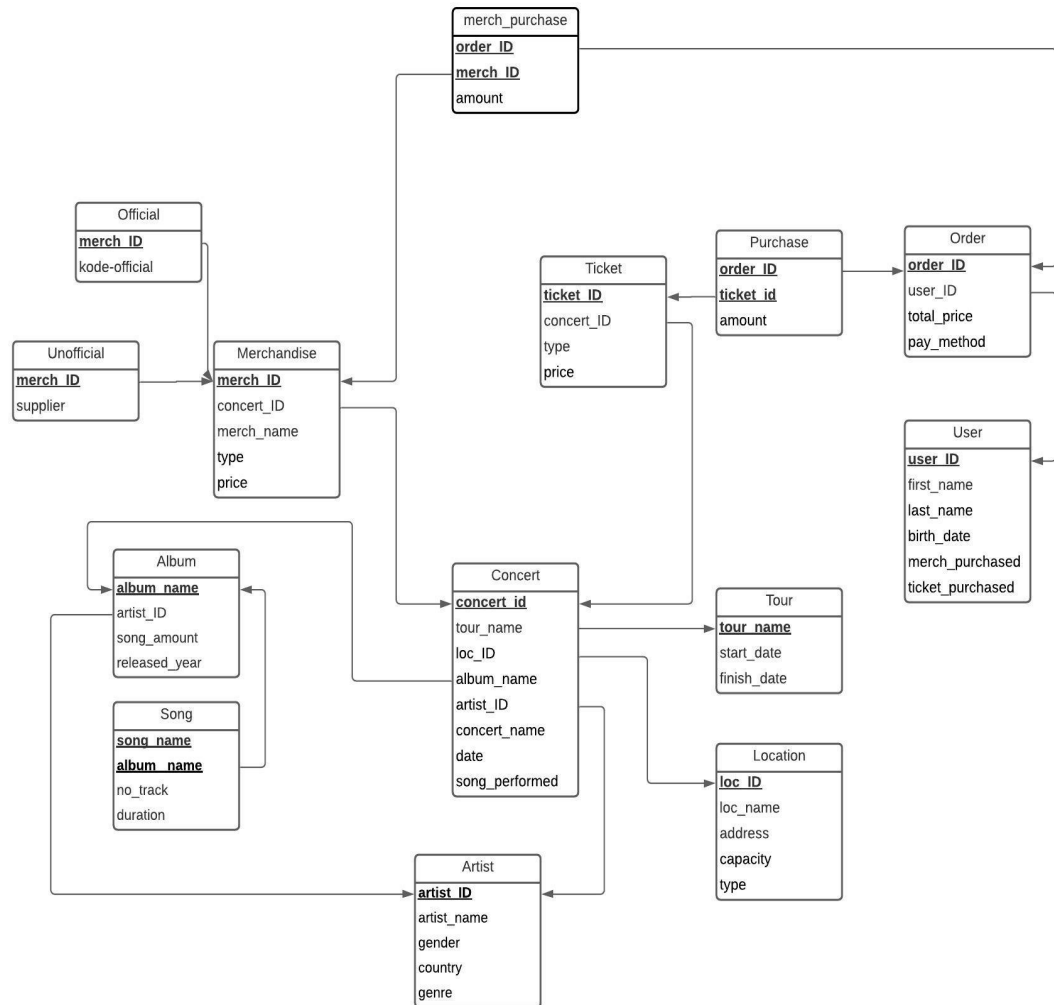
# 1. Desain Basis Data Relasional

## a. Skema Basis Data



**Skema Awal Basis Data**

Berkaitan dengan implementasi *Business Rule* yang akan dilakukan nantinya, maka akan perlu ditambahkan relasi yang mencatat mengenai transaksi *merchandise* yang pernah dilakukan pengguna. Hal ini tidak diimplementasikan sesuai dengan deskripsi kebutuhan yang disebutkan sebelumnya. Dengan menambahkan relasi '*merch\_purchase*', relasi ini akan menangani pembelian *merchandise* untuk setiap pemesanan. Berikut merupakan skema basis data yang merupakan hasil revisi tersebut.



### Skema Basis Data Revisi

Pada skema di atas, dapat terlihat sebuah relasi transaksi baru '*merch\_purchase*' yang mencatat pembelian *merchandise* oleh pengguna di suatu waktu. Selain itu, beberapa perubahan untuk menyesuaikan skema basis data awal dengan spesifikasi *business rule* yang diberikan adalah penambahan atribut '*merch\_purchased*' dan '*ticket\_purchased*' pada relasi *User* untuk mencatat banyak *merchandise* dan tiket yang sudah dibeli pengguna, serta '*song\_performed*' pada relasi *Concert* untuk mencatat banyak lagu yang dibawakan di suatu konser dan '*amount*' pada *Purchase* untuk mencatat jumlah tiket yang dibeli di suatu *order*.

## b. **Dependensi Fungsional**

Untuk relasi *Merchandise*, terdapat beberapa *functional dependency* yang terbentuk antara atribut-atribut pada relasi ini. Karena *left-hand side* atau determinan dari *functional dependency* yang terbentuk adalah sama, maka beberapa *functional dependency* ini dapat digabung. Dengan demikian, dihasilkan *functional dependency* '*merch\_ID*  $\rightarrow$  *concert\_ID, merch\_name, type, price*'. Hal ini dikarenakan suatu *merch\_ID* hanya memiliki tepat satu konser tempat penjualannya, nama *merchandise*, serta tipe dan harga dari *merchandise* ini.

Pada relasi *Official*, *functional dependency* yang terbentuk antara atribut-atribut pada relasi ini adalah '*merch\_ID*  $\rightarrow$  *kode\_official*'. Dependensi ini berlaku karena setiap *merchandise* yang ada di dalam relasi ini harus tepat memiliki satu kode *official* tertentu. Namun, setiap kode bisa dimiliki oleh lebih dari satu *merchandise*. Dengan demikian, hanya *functional dependencies* di atas yang terdefinisi untuk relasi *Official*.

Pada relasi *Unofficial*, *functional dependency* yang terbentuk antara atribut-atribut pada relasi ini adalah '*merch\_ID*  $\rightarrow$  *supplier*'. Dependensi ini berlaku karena setiap *merchandise* yang ada di dalam relasi ini harus tepat memiliki satu *supplier* tertentu. Namun, setiap *supplier* bisa jadi melakukan *supplying* ke lebih dari satu jenis *merchandise*. Dengan demikian, hanya ada *functional dependency* di atas pada relasi *Unofficial* ini.

Pada relasi *Merch\_Purchase*, *functional dependency* yang terbentuk antara atribut-atribut relasi ini adalah '*order\_ID, merch\_ID*  $\rightarrow$  *amount*'. Dependensi ini berlaku karena setiap pembelian *merchandise* yang ada di suatu *order* tertentu akan memiliki tepat satu buah jumlah *merchandise* yang dibeli. Dengan demikian, hanya ada *functional dependency* tersebut untuk relasi *Merch\_Purchase* tersebut.

Pada relasi *Song*, *functional dependency* yang terbentuk antara atribut-atribut pada relasi tersebut adalah sebagai berikut.

$$F = \{\text{song\_name, album\_name} \rightarrow \text{no\_track, duration}\}$$

Dependensi '*song\_name, album\_name*  $\rightarrow$  *no\_track, duration*' berlaku karena satu nilai atribut *song\_name* dan *album\_name* hanya berpasangan dengan satu nilai *no\_track* dan satu nilai *duration*.

Pada relasi *Album*, *functional dependency* yang terbentuk antara atribut-atribut pada relasi tersebut adalah sebagai berikut.

$$F = \{\text{album\_name} \rightarrow \text{artist\_ID, song\_amount, released\_year}\}$$

Dependensi '*album\_name*  $\rightarrow$  *artist\_ID*, *song\_amount*, *released\_year*' berlaku karena satu nilai atribut *album\_name* hanya berpasangan pada satu nilai *artist\_ID*, satu nilai *song\_amount*, dan satu nilai *released\_year*.

Pada relasi *Artist*, *functional dependency* yang terbentuk antara atribut-atribut pada relasi tersebut adalah sebagai berikut.

$$F = \{ \text{artist\_ID} \rightarrow \text{artist\_name, gender, country, genre} \}$$

Dependensi '*artist\_ID*  $\rightarrow$  *artist\_name*, *gender*, *country*, *genre*' berlaku karena satu nilai atribut *artist\_ID* hanya berpasangan pada satu nilai *artist\_name*, satu nilai *gender*, satu nilai *country*, dan satu nilai *genre*.

Pada relasi *Tour*, *functional dependency* yang terbentuk antara atribut-atribut pada relasi tersebut adalah,

$$F = \{ \text{tour\_name} \rightarrow \text{start\_date, finish\_date} \}$$

Dependensi '*tour\_name*  $\rightarrow$  *start\_date*, *finish\_date*' berlaku karena satu nilai atribut *tour\_name* hanya berpasangan pada satu nilai *start\_date* dan satu nilai *finish\_date*.

Pada relasi *Location*, *functional dependency* yang terbentuk antara atribut-atribut pada relasi tersebut adalah,

$$F = \{ \text{loc\_ID} \rightarrow \text{loc\_name, address, capacity, type} \}$$

Dependensi '*loc\_ID*  $\rightarrow$  *loc\_name*, *address*, *capacity*, *type*' berlaku karena satu nilai atribut *loc\_ID* hanya berpasangan pada satu nilai *loc\_name*, satu nilai *address*, satu nilai *capacity*, dan satu nilai *type*.

Pada relasi *Concert*, *functional dependency* yang terbentuk antara atribut-atribut pada relasi tersebut adalah,

$$F = \{ \text{concert\_ID} \rightarrow \text{tour\_name, loc\_ID, album\_name, artist\_ID, concert\_name, date, song\_performed} \}$$

Dependensi '*concert\_ID*  $\rightarrow$  *tour\_name*, *loc\_ID*, *album\_name*, *artist\_ID*, *concert\_name*, *date*, *song\_performed*' berlaku karena satu nilai atribut *concert\_ID* hanya berpasangan pada satu nilai *tour\_name*, satu nilai *loc\_ID*, satu nilai *album\_name*, satu nilai *artist\_ID*, satu nilai *concert\_name*, satu nilai *date*, dan satu nilai *song\_performed*.

Pada relasi *Ticket*, *functional dependency* yang terbentuk antara atribut-atribut pada relasi tersebut adalah,

$$F = \{\text{ticket\_ID} \rightarrow \text{concert\_ID}, \text{type}, \text{price}\}$$

Dependency di atas berlaku karena suatu nilai *ticket\_ID* hanya memiliki sebuah nilai *concert\_ID*, satu *type*, dan satu nilai *price*.

Pada relasi *User*, *functional dependency* yang terbentuk antara atribut-atribut pada relasi tersebut adalah,

$$F = \{\text{user\_ID} \rightarrow \text{first\_name}, \text{last\_name}, \text{birth\_date}, \text{ticket\_purchased}, \text{merch\_purchased}\}$$

Dependency di atas berlaku karena suatu nilai *user\_ID* hanya memiliki sebuah nilai *first\_name*, *last\_name*, *birth\_date*, *ticket\_purchased*, dan *merch\_purchased*.

Pada relasi *Order*, *functional dependency* yang terbentuk antara atribut-atribut pada relasi tersebut adalah sebagai berikut

$$F = \{\text{order\_ID} \rightarrow \text{user\_ID}, \text{total\_price}, \text{pay\_method}\}$$

Dependensi '*order\_ID*  $\rightarrow$  *user\_ID*, *total\_price*, *pay\_method*' berlaku karena satu nilai atribut *order\_ID* hanya berpasangan pada satu nilai *user\_ID*.

Pada relasi *Purchase*, *functional dependency* yang terbentuk antara atribut-atribut pada relasi tersebut adalah sebagai berikut

$$F = \{\text{order\_ID}, \text{ticket\_ID} \rightarrow \text{amount}\}$$

Dependensi '*order\_ID*, *ticket\_ID*  $\rightarrow$  *amount*' berlaku karena dua nilai atribut *order\_ID* dan *ticket\_ID* yang berpasangan pada satu nilai *order\_ID* dari relasi *Order* dan *ticket\_ID* dari relasi *Ticket*.

### c. Identifikasi Bentuk Normal Relasi

Pada relasi *Merchandise* terdefinisi *functional dependency* bahwa '*merch\_ID* *determines* *concert\_ID*, *merch\_name*, *type*, *price*'. Untuk relasi ini maka dapat kita ketahui bahwa *candidate key* sekaligus *primary key* dari relasi ini adalah '*merch\_ID*'. Karena hanya terdefinisi sebuah dependensi yang determinannya adalah *primary key* relasi tersebut, maka dapat dikatakan bahwa relasi *merchandise* ini telah berada dalam bentuk normal BCNF.

Untuk relasi *Official*, terdefinisi sebuah dependensi berupa '*merch\_ID*  $\rightarrow$  *kode\_official*'. Dari atribut-atribut yang ada dapat diketahui bahwa '*merch\_ID*' juga merupakan *candidate key* untuk relasi ini. Dengan demikian, relasi *official* sudah berada dalam bentuk normal BCNF karena determinan dependensi fungsionalnya merupakan *primary* atau *candidate key*.

Hal serupa juga terjadi pada relasi *Unofficial*, yang hanya memiliki satu dependensi fungsional, yaitu ' $\text{merch\_ID} \rightarrow \text{supplier}$ '. Untuk relasi ini, atribut ' $\text{merch\_ID}$ ' lagi-lagi menjadi *candidate* sekaligus *primary key*. Dengan demikian, maka relasi *Unofficial* telah berada dalam bentuk normal BCNF karena determinan dependensi fungsionalnya merupakan *primary* atau *candidate key*.

Pada relasi *Merch\_Purchase*, terdefinisi dependensi fungsional berupa ' $\text{order\_ID}, \text{merch\_ID} \rightarrow \text{amount}$ '. Untuk relasi ini dapat diketahui bahwa kombinasi antara ' $\text{order\_ID}$ ' dan ' $\text{merch\_ID}$ ' merupakan *candidate* sekaligus *primary key*. Karena *left hand side* dari semua dependensinya merupakan *candidate key*, maka relasi ini masuk ke dalam bentuk normal BCNF.

Pada relasi *Song*, terdefinisi satu *functional dependency*: ' $\text{song\_name}, \text{album\_name} \rightarrow \text{no\_track}, \text{duration}$ '. Kunci kandidat pada relasi *Song* yaitu kombinasi *song\_name* dan *album\_name*. Karena semua *functional dependency* pada *Song* memiliki *left hand side* berupa kunci kandidat, relasi *Song* memiliki bentuk normal BCNF.

Pada relasi *Album*, terdefinisi satu *functional dependency*: ' $\text{album\_name} \rightarrow \text{artist\_ID}, \text{song\_amount}, \text{released\_year}$ '. Kunci kandidat pada relasi *Album* yaitu *album\_name*. Karena semua *functional dependency* pada *Album* memiliki *left hand side* berupa kunci kandidat, relasi *Album* memiliki bentuk normal BCNF.

Pada relasi *Artist*, terdefinisi satu *functional dependency*: ' $\text{artist\_ID} \rightarrow \text{artist\_name}, \text{gender}, \text{country}, \text{genre}$ '. Kunci kandidat pada relasi *Artist* yaitu *artist\_ID*. Karena semua *functional dependency* pada *Artist* memiliki *left hand side* berupa kunci kandidat, relasi *Artist* memiliki bentuk normal BCNF.

Pada relasi *Tour*, terdefinisi satu *functional dependency* ' $\text{tour\_name} \rightarrow \text{start\_date}, \text{finish\_date}$ '. Kunci kandidat pada relasi *Tour* yaitu *tour\_name*. Karena semua *functional dependency* pada *Tour* memiliki *left hand side* berupa kunci kandidat, relasi *Tour* memiliki bentuk normal BCNF.

Pada relasi *Location*, terdefinisi satu *functional dependency* ' $\text{loc\_ID} \rightarrow \text{loc\_name}, \text{address}, \text{capacity}, \text{type}$ '. Kunci kandidat pada relasi *Location* yaitu *loc\_ID*. Karena semua *functional dependency* pada *Location* memiliki *left hand side* berupa kunci kandidat, relasi *Location* memiliki bentuk normal BCNF.

Pada relasi *Concert*, terdefinisi satu *functional dependency* ' $\text{concert\_ID} \rightarrow \text{tour\_name}, \text{loc\_ID}, \text{album\_name}, \text{artist\_ID}, \text{concert\_name}, \text{date}, \text{song\_performed}$ '. Kunci kandidat pada relasi *Concert* yaitu *concert\_ID*. Karena semua *functional dependency* pada *Concert* memiliki *left hand side* berupa kunci kandidat, relasi *Concert* memiliki bentuk normal BCNF.

Pada relasi *Ticket*, terdefinisi hanya sebuah *functional dependency*, yaitu ' $ticket\_ID \rightarrow concert\_ID, type, price$ '. Pada kasus ini, *ticket\_ID* menentukan nilai dari seluruh atribut relasi lainnya. Karena itu, *ticket\_ID* menjadi *candidate key* untuk relasi ini. Karena *left hand side* dari seluruh *functional dependency* relasi merupakan kunci kandidat, relasi ini dapat dikategorikan dalam bentuk normal BCNF.

Pada relasi *User*, terdefinisi hanya satu *functional dependency*, yaitu ' $user\_ID \rightarrow first\_name, last\_name, birth\_date, ticket\_purchased, merch\_purchased$ '. Pada kasus ini, *user\_ID* menentukan nilai dari seluruh atribut relasi lainnya. Karena itu, *user\_ID* menjadi *candidate key* untuk relasi ini. Karena *left hand side* dari seluruh *functional dependency* relasi merupakan kunci kandidat, relasi ini dapat dikategorikan dalam bentuk normal BCNF.

Pada relasi *Order*, terdefinisi satu *functional dependency* ' $order\_ID \rightarrow user\_ID, total\_price, method$ '. Kunci kandidat pada relasi *Order* yaitu *order\_ID*. Karena semua *functional dependency* pada *Order* memiliki *left hand side* berupa kunci kandidat, relasi *Order* memiliki bentuk normal BCNF.

Pada relasi *Purchase*, terdefinisi satu *functional dependency* ' $order\_ID, ticket\_ID \rightarrow amount$ '. Untuk relasi ini dapat diketahui bahwa kombinasi antara ' $order\_ID$ ' dan ' $ticket\_ID$ ' merupakan *candidate* sekaligus *primary key*. Karena *left hand side* dari semua dependensinya merupakan *candidate key*, maka relasi ini masuk ke dalam bentuk normal BCNF.

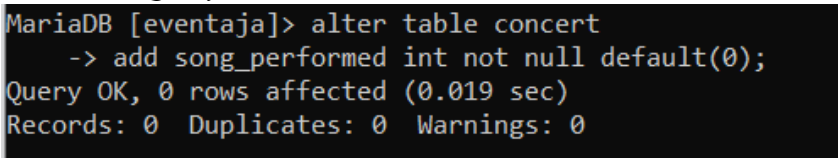
#### **d. Normalisasi Skema Basis Data**

Untuk relasi *Merchandise*, *Official*, *Unofficial*, *Merch\_Purchase*, *Song*, *Album*, *Artist*, *Tour*, *Location*, *Concert*, *Ticket*, *User*, *Order*, dan *Purchase* tidak perlu dilakukan normalisasi lagi untuk ketiga relasi ini. Hal ini dikarenakan semua relasi yang telah didaftarkan di atas sudah berada dalam bentuk normal BCNF sehingga relasi ini dapat dikatakan sudah memiliki desain yang cukup baik, terlepas dari *dependency preserving* atau tidaknya relasi ini terhadap atribut-atributnya.

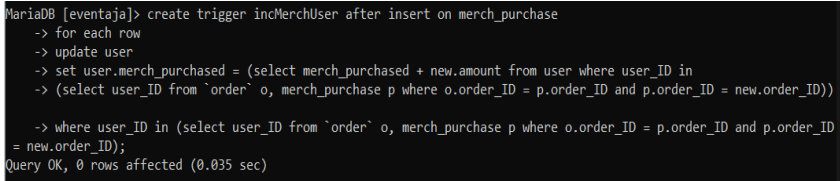


## 2. Implementasi Business Rule

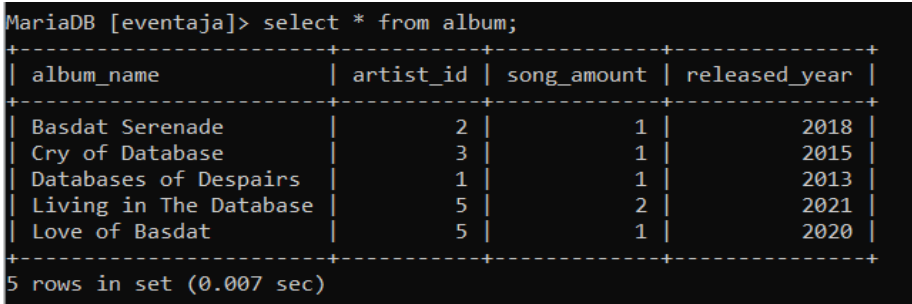
### a. Implementasi *Business Rule*

<i>Business Rule</i>	<i>Implementasi Query</i>
Perhitungan banyak lagu yang dinyanyikan pada suatu konser dan memastikan lagu yang dibawakan tidak lebih banyak dari jumlah lagu yang terdapat pada album.	<p>Penambahan atribut <code>'song_performed'</code> pada <i>Concert</i>:</p> <pre>alter table concert add song_performed int not null default(0);</pre> <p>Implementasi <i>Trigger</i> untuk <i>Update</i> pada <i>Concert</i>:</p> <pre>delimiter - create trigger check_upd_perform before update on concert for each row begin if((select      song_amount      from      album      where album.album_name      =      new.album_name)      &lt; new.song_performed) then signal sqlstate '45000' set message_text = 'Invalid amount (bigger than the number of song in the album) of song updated into the concert!'; end if; end; - delimiter ;</pre> <p>Implementasi <i>Trigger</i> untuk <i>Insert</i> pada <i>Concert</i>:</p> <pre>delimiter - create trigger check_ins_perform before insert on concert for each row begin if((select      song_amount      from      album      where album.album_name      =      new.album_name)      &lt; new.song_performed) then signal sqlstate '45000' set message_text = 'Invalid amount (bigger than the number of song in the album) of song inserted into the concert!'; end if; end; - delimiter ;</pre> <p><i>Screenshot Query:</i></p>  <pre>MariaDB [eventaja]&gt; alter table concert -&gt; add song_performed int not null default(0); Query OK, 0 rows affected (0.019 sec) Records: 0 Duplicates: 0 Warnings: 0</pre>

	<pre> MariaDB [eventaja]&gt; delimiter - MariaDB [eventaja]&gt; create trigger check_upd_perform before update on concert -&gt; for each row -&gt; begin -&gt; if((select song_amount from album where album.album_name = new.album_name) &lt; new.song_performed) -&gt; then -&gt; signal sqlstate '45000' -&gt; set message_text = 'Invalid amount (bigger than the number of song in the album) of song updated into the concert!'; -&gt; end if; -&gt; end; - Query OK, 0 rows affected (0.019 sec)  MariaDB [eventaja]&gt; delimiter ; MariaDB [eventaja]&gt; delimiter - MariaDB [eventaja]&gt; create trigger check_ins_perform before insert on concert -&gt; for each row -&gt; begin -&gt; if((select song_amount from album where album.album_name = new.album_name) &lt; new.song_performed) -&gt; then -&gt; signal sqlstate '45000' -&gt; set message_text = 'Invalid amount (bigger than the number of song in the album) of song inserted into the concert!'; -&gt; end if; -&gt; end; - Query OK, 0 rows affected (0.042 sec) </pre>
Penambahan total harga yang harus dibayar pada pesanan (order)	<p><b>Implementasi <i>Trigger</i>:</b></p> <pre> create trigger totalOrderPrice_insticPurc after insert on purchase for each row update `order` set total_price = total_price + ( select sum(Ticket.price * new.amount) from Ticket where Ticket.ticket_ID = new.ticket_ID ) where new.order_ID = order_ID;  create trigger totalOrderPrice_insMercPurc after insert on merch_purchase for each row update `order` set total_price = total_price + ( select sum(merchandise.price * new.amount) from merchandise where merchandise.merch_ID = new.merch_ID ) where new.order_ID = order_ID; </pre>
Perhitungan <i>Merchandise</i> yang telah dibeli pengguna.	<p><b>Implementasi <i>Trigger</i>:</b></p> <pre> create trigger incMerchUser after insert on merch_purchase for each row update user set user.merch_purchased = (select merch_purchased + new.amount from user where user_ID in (select user_ID from `order` o, merch_purchase p where o.order_ID = p.order_ID and p.order_ID = new.order_ID)) </pre>

	<pre>where user_ID in (select user_ID from `order` o, merch_purchase p where o.order_ID = p.order_ID and p.order_ID = new.order_ID);</pre> <p><b>Screenshot Query:</b></p> 
Perhitungan Ticket yang telah dibeli pengguna	<p><b>Implementasi Trigger:</b></p> <pre>create trigger incTicketUser after insert on purchase for each row update user set user.ticket_purchased = (select ticket_purchased+new.amount from user where user_ID in (select user_ID from `order` o,purchase p where o.order_ID = p.order_ID and p.order_ID = new.order_ID)) where user_ID in (select user_ID from `order` o,purchase p where o.order_ID = p.order_ID and p.order_ID = new.order_ID);</pre>

## b. Aplikasi *Business Rule*

<i>Business Rule</i>	<i>Aplikasi Query</i>
Perhitungan banyak lagu yang dinyanyikan pada suatu konser dan memastikan lagu yang dibawakan tidak lebih banyak dari jumlah lagu yang terdapat pada album.	<p><b>Aplikasi Query:</b></p> <p>Berikut merupakan tabel <i>Album</i> pada basis data:</p>  <p>Pada saat dilakukan <i>update</i> dengan jumlah yang melebihi jumlah lagu pada album yang berkaitan, misalnya dengan melakukan <i>update</i> pada tabel <i>Concert</i> untuk album <i>Love of Basdat</i> dan mengisi jumlah lagu</p>

yang dibawakan sebanyak 2, maka akan terjadi *error* seperti pada *query* berikut:

*Query*:

```
update concert
set song_performed = 2
where album_name = "Love of Basdat";
```

```
MariaDB [eventaja]> update concert
-> set song_performed = 2
-> where album_name = "Love of Basdat";
ERROR 1644 (45000): Invalid amount (bigger than the number of song in the album) of song updated into the concert!
MariaDB [eventaja]>
```

Kasus lainnya adalah misalkan ada penambahan konser baru untuk album *Basdat Serenade* dan di konser ini akan dibawakan 5 lagu. Maka, dengan demikian akan terjadi *error* seperti pada *query* berikut:

*Query*:

```
insert into concert
(tour_name, loc_ID, album_name, artist_ID, concert_name,
date, song_performed)
values("Invalid", 2, "Basdat Serenade", 2, "Invalid
Concert", '2005-06-29', 5);
```

```
MariaDB [eventaja]> insert into concert
-> (tour_name, loc_ID, album_name, artist_ID, concert_name, date, song_performed)
-> values("Invalid", 2, "Basdat Serenade", 2, "Invalid Concert", '2005-06-29', 5);
ERROR 1644 (45000): Invalid amount (bigger than the number of song in the album) of song inserted into the
concert!
```

Namun, apabila jumlah lagu yang dibawakan adalah valid, dalam hal ini kurang dari atau sama dengan jumlah lagu yang ada pada album terkait, maka *update* dan *insert* akan berjalan lancar, seperti pada *Query* berikut:

*Query*:

```
update concert
set song_performed = 1
where album_name = "Love of Basdat";
```

```
insert into concert
(tour_name, loc_ID, album_name, artist_ID, concert_name,
date, song_performed)
values("Basis", 2, "Basdat Serenade", 2, "Invalid
Concert", '2005-06-29', 1);
```

```
select * from concert;
```

```
MariaDB [eventaja]> update concert
-> set song_performed = 1
-> where album_name = "Love of Basdat";
Query OK, 2 rows affected (0.009 sec)
Rows matched: 2 Changed: 2 Warnings: 0
```

	<pre>MariaDB [eventaja]&gt; insert into concert -&gt; (tour_name, loc_ID, album_name, artist_ID, concert_name, date, song_performed) -&gt; values("Basis", 2, "Basdat Serenade", 2, "Invalid Concert", '2005-06-29', 1); Query OK, 1 row affected (0.008 sec)</pre> <pre>MariaDB [eventaja]&gt; select * from concert;</pre> <table><tr><th>concert_id</th><th>tour_name</th><th>loc_ID</th><th>album_name</th><th>artist_ID</th><th>concert_name</th><th>date</th><th>song_performed</th></tr><tr><td>1</td><td>Basis</td><td>1</td><td>Basdat Serenade</td><td>1</td><td>Basisten</td><td>2021-03-17</td><td>0</td></tr><tr><td>2</td><td>Lilacs</td><td>2</td><td>Cry of Database</td><td>2</td><td>Lilacsin</td><td>2021-02-17</td><td>0</td></tr><tr><td>3</td><td>Roses</td><td>3</td><td>Databases of Despairs</td><td>3</td><td>Databases My Fav</td><td>2021-02-01</td><td>0</td></tr><tr><td>4</td><td>Tulips</td><td>4</td><td>Living in The Database</td><td>4</td><td>Livingers</td><td>2021-07-10</td><td>0</td></tr><tr><td>5</td><td>Violets</td><td>5</td><td>Love of Basdat</td><td>5</td><td>My Love</td><td>2021-02-14</td><td>1</td></tr><tr><td>6</td><td>Basis</td><td>2</td><td>Basdat Serenade</td><td>1</td><td>Basisten</td><td>2021-03-21</td><td>0</td></tr><tr><td>7</td><td>Basis</td><td>5</td><td>Basdat Serenade</td><td>1</td><td>Basisten</td><td>2021-03-25</td><td>0</td></tr><tr><td>8</td><td>Tulips</td><td>4</td><td>Love of Basdat</td><td>5</td><td>My Love</td><td>2021-02-16</td><td>1</td></tr><tr><td>10</td><td>Basis</td><td>2</td><td>Basdat Serenade</td><td>2</td><td>Invalid Concert</td><td>2005-06-29</td><td>1</td></tr></table> <pre>9 rows in set (0.001 sec)</pre>	concert_id	tour_name	loc_ID	album_name	artist_ID	concert_name	date	song_performed	1	Basis	1	Basdat Serenade	1	Basisten	2021-03-17	0	2	Lilacs	2	Cry of Database	2	Lilacsin	2021-02-17	0	3	Roses	3	Databases of Despairs	3	Databases My Fav	2021-02-01	0	4	Tulips	4	Living in The Database	4	Livingers	2021-07-10	0	5	Violets	5	Love of Basdat	5	My Love	2021-02-14	1	6	Basis	2	Basdat Serenade	1	Basisten	2021-03-21	0	7	Basis	5	Basdat Serenade	1	Basisten	2021-03-25	0	8	Tulips	4	Love of Basdat	5	My Love	2021-02-16	1	10	Basis	2	Basdat Serenade	2	Invalid Concert	2005-06-29	1
concert_id	tour_name	loc_ID	album_name	artist_ID	concert_name	date	song_performed																																																																										
1	Basis	1	Basdat Serenade	1	Basisten	2021-03-17	0																																																																										
2	Lilacs	2	Cry of Database	2	Lilacsin	2021-02-17	0																																																																										
3	Roses	3	Databases of Despairs	3	Databases My Fav	2021-02-01	0																																																																										
4	Tulips	4	Living in The Database	4	Livingers	2021-07-10	0																																																																										
5	Violets	5	Love of Basdat	5	My Love	2021-02-14	1																																																																										
6	Basis	2	Basdat Serenade	1	Basisten	2021-03-21	0																																																																										
7	Basis	5	Basdat Serenade	1	Basisten	2021-03-25	0																																																																										
8	Tulips	4	Love of Basdat	5	My Love	2021-02-16	1																																																																										
10	Basis	2	Basdat Serenade	2	Invalid Concert	2005-06-29	1																																																																										
Perhitungan total_price	<p>Aplikasi <i>Query</i>:</p> <p>Memasukkan order baru dalam relasi.</p> <pre>insert into `order`(user_ID,pay_method,total_price) values (2,"Gopay",0);</pre> <p>Order baru terletak di baris paling bawah dengan <i>order_ID</i> bernilai 11 dan <i>user_ID</i> bernilai 2.</p> <p>Menampilkan order baru tersebut dalam tabel order.</p> <pre>MariaDB [eventaja]&gt; insert into `order`(user_ID,pay_method,total_price) values(2,"Gopay",0); Query OK, 1 row affected (0.076 sec)</pre> <pre>MariaDB [eventaja]&gt; select * from `order`;</pre> <table><tr><th>order_ID</th><th>user_ID</th><th>total_price</th><th>pay_method</th></tr><tr><td>1</td><td>1</td><td>100000</td><td>Gopay</td></tr><tr><td>2</td><td>2</td><td>100000</td><td>M-Banking</td></tr><tr><td>3</td><td>3</td><td>200000</td><td>DANA</td></tr><tr><td>4</td><td>4</td><td>300000</td><td>ATM</td></tr><tr><td>5</td><td>5</td><td>200000</td><td>DANA</td></tr><tr><td>6</td><td>2</td><td>1100000</td><td>Gopay</td></tr><tr><td>7</td><td>1</td><td>400000</td><td>M-Banking</td></tr><tr><td>8</td><td>3</td><td>700000</td><td>DANA</td></tr><tr><td>9</td><td>5</td><td>700000</td><td>ATM</td></tr><tr><td>10</td><td>2</td><td>700000</td><td>M-Banking</td></tr><tr><td>11</td><td>2</td><td>0</td><td>Gopay</td></tr></table> <pre>11 rows in set (0.001 sec)</pre> <pre>MariaDB [eventaja]&gt;</pre> <p>Memasukkan data pembelian merchandise dengan <i>order_ID</i> bernilai 11</p> <pre>insert into merch_purchase(order_ID, merch_ID, amount) values (11, 2, 2);</pre> <p>Menampilkan perubahan pada tabel <i>order</i>.</p>	order_ID	user_ID	total_price	pay_method	1	1	100000	Gopay	2	2	100000	M-Banking	3	3	200000	DANA	4	4	300000	ATM	5	5	200000	DANA	6	2	1100000	Gopay	7	1	400000	M-Banking	8	3	700000	DANA	9	5	700000	ATM	10	2	700000	M-Banking	11	2	0	Gopay																																
order_ID	user_ID	total_price	pay_method																																																																														
1	1	100000	Gopay																																																																														
2	2	100000	M-Banking																																																																														
3	3	200000	DANA																																																																														
4	4	300000	ATM																																																																														
5	5	200000	DANA																																																																														
6	2	1100000	Gopay																																																																														
7	1	400000	M-Banking																																																																														
8	3	700000	DANA																																																																														
9	5	700000	ATM																																																																														
10	2	700000	M-Banking																																																																														
11	2	0	Gopay																																																																														

```
MariaDB [eventaja]> insert into merch_purchase(order_ID, merch_ID, amount)
-> values(11, 2, 2);
Query OK, 1 row affected (0.029 sec)

MariaDB [eventaja]> select * from `order`;
+-----+-----+-----+-----+
| order_ID | user_ID | total_price | pay_method |
+-----+-----+-----+-----+
| 1 | 1 | 100000 | Gopay |
| 2 | 2 | 100000 | M-Banking |
| 3 | 3 | 200000 | DANA |
| 4 | 4 | 300000 | ATM |
| 5 | 5 | 200000 | DANA |
| 6 | 2 | 1100000 | Gopay |
| 7 | 1 | 400000 | M-Banking |
| 8 | 3 | 700000 | DANA |
| 9 | 5 | 700000 | ATM |
| 10 | 2 | 700000 | M-Banking |
| 11 | 2 | 900000 | Gopay |
+-----+-----+-----+-----+
11 rows in set (0.001 sec)
```

Memasukkan data pembelian *ticket* dengan *order\_ID* bernilai 11.

```
insert into purchase(order_ID, ticket_id, amount) values
(11,6,3);
```

Menampilkan perubahan pada tabel *order*.

```
MariaDB [eventaja]> insert into purchase(order_ID, ticket_id, amount) values (11,6,3);
Query OK, 1 row affected (0.049 sec)

MariaDB [eventaja]> select * from `order`;
+-----+-----+-----+-----+
| order_ID | user_ID | total_price | pay_method |
+-----+-----+-----+-----+
| 1 | 1 | 100000 | Gopay |
| 2 | 2 | 100000 | M-Banking |
| 3 | 3 | 200000 | DANA |
| 4 | 4 | 300000 | ATM |
| 5 | 5 | 200000 | DANA |
| 6 | 2 | 1100000 | Gopay |
| 7 | 1 | 400000 | M-Banking |
| 8 | 3 | 700000 | DANA |
| 9 | 5 | 700000 | ATM |
| 10 | 2 | 700000 | M-Banking |
| 11 | 2 | 1650000 | Gopay |
+-----+-----+-----+-----+
11 rows in set (0.001 sec)
```

Berdasarkan tampilan setelah penambahan data pembelian merchandise, didapat terjadi penambahan nilai *total\_price* dari 0 menjadi 900000. Ini dikarenakan pembelian merchandise dengan merch\_ID bernilai 2 sebanyak 2 buah menyebabkan penambahan sebanyak 900000 ( $2 * 450000$ ).

Berikut merupakan data *merchandise* beserta harganya. Dapat dilihat bahwa merchandise dengan merch\_ID bernilai 2 memiliki nilai *price* sebesar 450000.

```
MariaDB [eventaja]> select * from merchandise;
```

merch_ID	concert_ID	merch_name	type	price
1	1	Stickers	unofficial	10000
2	1	Patches	official	450000
3	2	Guitar picks	official	25000
4	3	Keychains	official	35000
5	4	Tote Bags	unofficial	55000
6	5	Patches	unofficial	225000
7	2	Beanies Hat	official	150000
8	3	Guitar picks	unofficial	20000
9	4	Posters	official	35000
10	5	Posters	official	55000
11	5	Tote Bags	unofficial	45000
12	6	Tote Bags	unofficial	100000

```
12 rows in set (0.001 sec)
```

Berdasarkan setelah data pembelian, didapat perubahan *total\_price* sebesar 750000 (1650000 - 900000). Perubahan harga disebabkan adanya tambahan pembelian *ticket* dengan *ticket\_ID* bernilai 6 dan yang harga satuannya sebesar 250000 sebanyak 3 buah (3 \* 250000).

Berikut merupakan data *ticket* beserta harganya. Dapat dilihat bahwa *ticket* dengan *ticket\_ID* bernilai 6 memiliki harga sebesar 250000.

```
MariaDB [eventaja]> select * from ticket;
```

ticket_ID	concert_ID	type	price
1	1	Silver	100000
2	2	Silver	100000
3	3	Gold	200000
4	4	Premium	300000
5	5	Gold	200000
6	6	Silver	250000
7	6	Gold	300000
8	6	VIP	400000
9	7	Silver	400000
10	7	Gold	650000
11	7	VIP	900000
12	8	Festival	700000
13	8	Premium	1100000
14	8	Premium	1100000
15	8	Premium	1100000
16	8	Premium	1100000
17	8	Festival	700000
18	8	Festival	700000
19	8	Festival	700000

```
19 rows in set (0.001 sec)
```

Perhitungan *Merchandise* yang telah dibeli pengguna.

Aplikasi *Query*:  
Memasukkan data pembelian *merchandise*, *timestamp* secara *default* adalah waktu terjadinya *query*.

```
insert into merch_purchase(order_ID, merch_ID, amount)
values(1, 2, 2),
(5, 2, 5),
(2, 1, 6),
(4, 12, 10);
```

Menampilkan perubahan pada tabel *User*.

```
select * from user;
```

Menampilkan perubahan pada tabel *merch\_purchase*,

```
select * from merch_purchase;
```

### Screenshot *Query*:

```

MariaDB [eventaja]> select * from user;
+-----+-----+-----+-----+-----+-----+
| user_ID | first_name | last_name | birth_date | merch_purchased | ticket_purchased |
+-----+-----+-----+-----+-----+-----+
| 1 | Fabi | Anandi | 2001-01-22 | 0 | 2 |
| 2 | Eja | Morteza | 2000-11-09 | 0 | 1 |
| 3 | Nadim | Amizah | 2005-06-29 | 0 | 4 |
| 4 | Kiya | Utama | 2001-09-10 | 0 | 2 |
| 5 | Rafli | Ananda | 2002-05-12 | 0 | 1 |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.001 sec)

MariaDB [eventaja]> insert into merch_purchase(order_ID, merch_ID, amount)
-> values(1, 2, 2),
-> (5, 2, 5),
-> (2, 1, 6),
-> (4, 12, 10);
Query OK, 4 rows affected (0.039 sec)
Records: 4 Duplicates: 0 Warnings: 0

MariaDB [eventaja]> select * from user;
+-----+-----+-----+-----+-----+-----+
| user_ID | first_name | last_name | birth_date | merch_purchased | ticket_purchased |
+-----+-----+-----+-----+-----+-----+
| 1 | Fabi | Anandi | 2001-01-22 | 2 | 2 |
| 2 | Eja | Morteza | 2000-11-09 | 6 | 1 |
| 3 | Nadim | Amizah | 2005-06-29 | 0 | 4 |
| 4 | Kiya | Utama | 2001-09-10 | 10 | 2 |
| 5 | Rafli | Ananda | 2002-05-12 | 5 | 1 |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.001 sec)

MariaDB [eventaja]> select * from merch_purchase;
+-----+-----+-----+
| order_ID | merch_ID | amount |
+-----+-----+-----+
| 1 | 2 | 2 |
| 2 | 1 | 6 |
| 4 | 12 | 10 |
| 5 | 2 | 5 |
+-----+-----+-----+
4 rows in set (0.001 sec)

```

Secara otomatis, ketika ada penambahan *purchase* pada tabel *merch\_purchase*, maka atribut '*merch\_purchase*' pada tabel *User* akan berubah pada pengguna terkait sesuai dengan jumlah yang ditambahkan.

Perhitungan Ticket yang telah dibeli pengguna

Sebelum dilakukan pemanggilan query *insert* :

```

MariaDB [eventaja]> select * from user;
+-----+-----+-----+-----+-----+-----+
| user_ID | first_name | last_name | birth_date | merch_purchased | ticket_purchased |
+-----+-----+-----+-----+-----+-----+
| 1 | Fabi | Anandi | 2001-01-22 | 2 | 0 |
| 2 | Eja | Morteza | 2000-11-09 | 6 | 0 |
| 3 | Nadim | Amizah | 2005-06-29 | 0 | 0 |
| 4 | Kiya | Utama | 2001-09-10 | 10 | 0 |
| 5 | Rafli | Ananda | 2002-05-12 | 5 | 0 |
+-----+-----+-----+-----+-----+-----+

MariaDB [eventaja]> select * from purchase;
Empty set (0.000 sec)

```

Proses *insert* data ke relasi *purchase* dan perubahan pada relasi *user* setelah data ditambahkan ke relasi *purchase* :



```
MariaDB [eventaja]> insert into purchase (order_ID,ticket_ID,amount) values (1,1,2),(2,2,1),(3,3,4),(4,4,2),(5,5,1);
Query OK, 5 rows affected (0.019 sec)
Records: 5 Duplicates: 0 Warnings: 0
```

```
MariaDB [eventaja]> select * from user;
```

user_ID	first_name	last_name	birth_date	merch_purchased	ticket_purchased
1	Fabi	Anandi	2001-01-22	2	2
2	Eja	Morteza	2000-11-09	6	1
3	Nadim	Amizah	2005-06-29	0	4
4	Kiya	Utama	2001-09-10	10	2
5	Rafli	Ananda	2002-05-12	5	1

```
5 rows in set (0.000 sec)
```

Atribut *ticket\_purchased* pada relasi *user* secara otomatis ditambahkan dengan nilai baru dari atribut *amount* pada relasi *purchase* untuk *user* yang bersesuaian.

### 3. Kesimpulan

Dari analisis mengenai *functional dependency*, bentuk normal, dan normalisasi relasi, dapat dikatakan bahwa skema basis data yang dibuat untuk membantu sistem *EventAja* sudah cukup baik. Dalam hal ini, dapat terlihat bahwa semua relasi yang dimiliki oleh basis data ini sudah berada dalam bentuk normal yang paling 'ketat', yaitu *Boyce-Codd Normal Form* (BCNF). Karena sudah masuk ke dalam bentuk normal BCNF, maka basis data ini dapat diputuskan untuk tidak dinormalisasi lebih lanjut.

Meskipun demikian, relasi yang demikian belum tentu memenuhi prinsip *dependency preserving*. Dengan demikian, bisa dikatakan bahwa normalisasi perlu dilakukan sesuai dengan kebutuhan dari basis data yang ingin dibuat, dan hal ini dikarenakan adanya *trade-off* yang dimiliki oleh bentuk-bentuk normal, misalnya antara BCNF dan 3NF.

Adapun implementasi *business rule* untuk basis data ini umumnya dilakukan dengan menggunakan *Trigger Function*. Hal ini dikarenakan perubahan yang disebabkan oleh penambahan *record* baru lebih cocok untuk menggunakan konsep *trigger* dibandingkan fungsi biasa. Dengan demikian, perubahan yang terjadi akibat insersi data baru akan dapat terlihat pada tabel atau relasi lain sesuai dengan *Trigger* yang telah dirancang.

### 4. Referensi

*Relational Database Design* oleh Tim Pengajar IF2140 2020/2021 Institut Teknologi Bandung, diakses pada 16 April 2021 pukul 20.15 WIB melalui [https://www.youtube.com/watch?v=-p0YX0Pq7wU&list=PLRdh21P3pZquoxKAx10i\\_kukXI98nCs6T](https://www.youtube.com/watch?v=-p0YX0Pq7wU&list=PLRdh21P3pZquoxKAx10i_kukXI98nCs6T).

*Relational Database Design* oleh Tim Pengajar IF2140 2020/2021 Institut Teknologi Bandung, diakses pada 16 April 2021 pukul 20.21 WIB melalui [https://stei19.kuliah.itb.ac.id/pluginfile.php/81106/mod\\_resource/content/2/IF2240%20-%20Relational%20Database%20Design.pdf](https://stei19.kuliah.itb.ac.id/pluginfile.php/81106/mod_resource/content/2/IF2240%20-%20Relational%20Database%20Design.pdf).

*Relational Database Schema Design Overview* oleh Kim Nguyen, diakses pada 16 April 2021 pukul 21.21 WIB melalui <https://medium.com/@kimtnguyen/relational-database-schema-design-overview-70e447ff66f9>.

## 5. Pembagian Tugas

A:

1. Tour, Location, Concert → 13519201
2. Song, Artist, Album → 13519175
3. Merch, Off, Unoff → 13519185
4. Ticket, User → 13519179
5. Order, Purchase → 13519199

B:

1. **13519199, 13519201** → 13519185
2. 13519175
3. Merch → 13519185  
Ticket → 13519179