//// Docs: https://dbml.dbdiagram.io/docs

//// -- LEVEL 1

//// -- Schemas, Tables and References

// Creating tables

// You can define the tables with full schema names

Table ecommerce.merchants {

  id int

  country\_code int

  merchant\_name varchar

  "created at" varchar

  admin\_id int [ref: > U.id]

  Indexes {

    (id, country\_code) [pk]

  }

}

// If schema name is omitted, it will default to "public" schema.

Table users as U {

  id int [pk, increment] // auto-increment

  full\_name varchar

  created\_at timestamp

  country\_code int

}

Table countries {

  code int [pk]

  name varchar

  continent\_name varchar

 }

// Creating references

// You can also define relaionship separately

// > many-to-one; < one-to-many; - one-to-one; <> many-to-many

Ref: U.country\_code > countries.code

Ref: ecommerce.merchants.country\_code > countries.code

//----------------------------------------------//

//// -- LEVEL 2

//// -- Adding column settings

Table ecommerce.order\_items {

  order\_id int [ref: > ecommerce.orders.id] // inline relationship (many-to-one)

  product\_id int

  quantity int [default: 1] // default value

}

Ref: ecommerce.order\_items.product\_id > ecommerce.products.id

Table ecommerce.orders {

  id int [pk] // primary key

  user\_id int [not null, unique]

  status varchar

  created\_at varchar [note: 'When order created'] // add column note

}

//----------------------------------------------//

//// -- Level 3

//// -- Enum, Indexes

// Enum for 'products' table below

Enum ecommerce.products\_status {

  out\_of\_stock

  in\_stock

  running\_low [note: 'less than 20'] // add column note

}

// Indexes: You can define a single or multi-column index

Table ecommerce.products {

  id int [pk]

  name varchar

  merchant\_id int [not null]

  price int

  status ecommerce.products\_status

  created\_at datetime [default: `now()`]

  Indexes {

    (merchant\_id, status) [name:'product\_status']

    id [unique]

  }

}

Table ecommerce.product\_tags {

  id int [pk]

  name varchar

}

Table ecommerce.merchant\_periods {

  id int [pk]

  merchant\_id int

  country\_code int

  start\_date datetime

  end\_date datetime

}

Ref: ecommerce.products.merchant\_id > ecommerce.merchants.id // many-to-one

Ref: ecommerce.product\_tags.id <> ecommerce.products.id // many-to-many

//composite foreign key

Ref: ecommerce.merchant\_periods.(merchant\_id, country\_code) > ecommerce.merchants.(id, country\_code)