**Tugas Web Database**

**Logical Data Design**

# D:\HIMTI\Hexion 2012\Logo\binus.png

**Dosen :** [Michael Yoseph Ricky, S.Kom., M.M](javascript:__doPostBack('ctl00$cpContent$listOfLecturer$repeaterListOfLecturer$ctl02$linkButtonNamaDosenAlternating',''))

**Kelas : 06 PFT**

Fery Kurniawan 1501146893

Edwin Anderson Loho 1501148942

David Lienardy 1501151855

Rama Adithya 1501164340

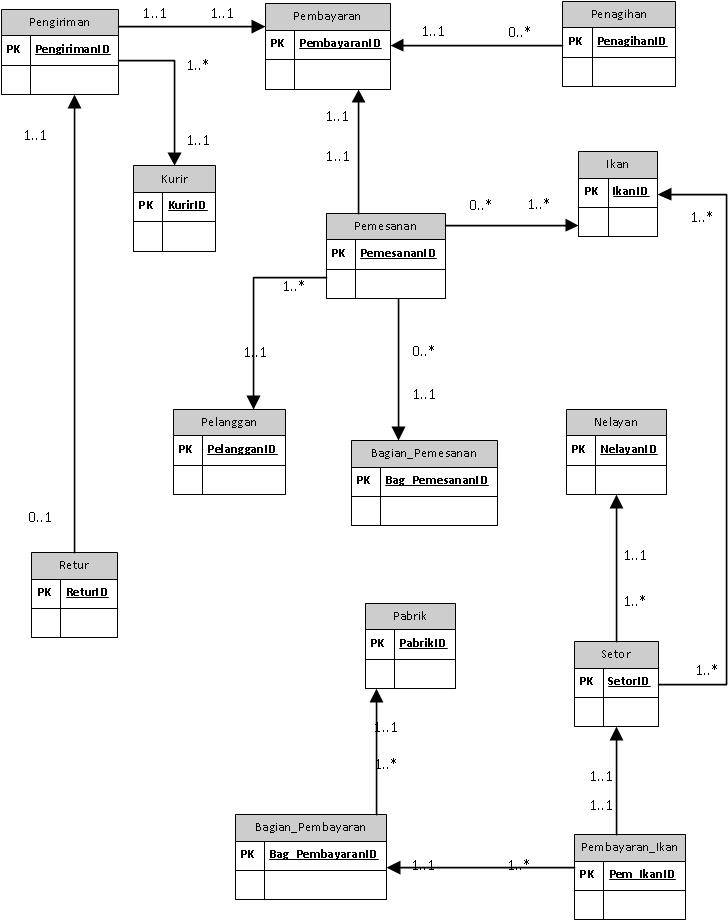
Katherine Oliviani 1501143752

**Universitas Bina Nusantara**

**Jakarta**

**2014**

**Model Data Konseptual**

****

**1. Menentukan Relasi Untuk Model Data Logikal**

Langkah ini bertujuan untuk menciptakan relasi untuk model data logikal yang mewakili entitas, *relationship*, dan atribut yang sudah diidentifikasi.

1. Identifikasi *Strong Entity Types*

*Strong entity* adalah *entity* yang tidak tergantung kepada entitas lainnya atau dengan kata lain entitas yang dapat berdiri sendiri. Berikut merupakan entitas yang tergolong kedalam *strong entity* :

1. Pelanggan (PelangganID, NamaPelanggan, AlamatPelanggan, DOB, Gender, PelangganTelpNum)

*Primary Key* PelangganID

1. Ikan(IkanID, NamaIkan, SellPrice, PurchasePrice, Weight)

*Primary Key* IkanID

1. Nelayan(NelayanID, NamaNelayan, AlamatNelayan, DOB, Gender, NelayanTelpNum)

*Primary Key* NelayanID

1. Bagian\_Pembayaran(Bag\_PembayaranID, NamaStaff, Jabatan, Alamat, Gaji)

*Primary Key* Bag\_PembayaranID

1. Pabrik(PabrikID, NamaPabrik, AlamatPabrik, TelpNumPabrik)

*Primary Key* PabrikID

1. Bagian\_Pemesanan (Bag\_PemesananID, NamaStaff, Jabatan, Alamat, Gaji)

*Primary Key* Bag\_PemesananID

1. Kurir(KurirID, NamaKurir)

*Primary Key* KurirID

1. *Weak Entity Types*

*Weak entity* adalah *entity* yang bergantung kepada entitas lainnya. Dengan kata lain, merupakan entitas yang tidak dapat berdiri sendiri. Berikut merupakan entitas yang tergolong kedalam *weak entity* :

1. Retur (ReturID, TanggalPengembalian, Alasan)

*Primary Key* ReturID

1. Pengiriman (PengirimanID, TanggalPengiriman)

*Primary Key* PengirimanID

1. Setor (SetorID, TanggalSetor)

*Primary Key* SetorID

1. Pembayaran (PembayaranID, TanggalPembayaran)

*Primary Key* PembayaranID

1. Penagihan (PenagihanID, TanggalPenagihan)

*Primary Key* PenagihanID

1. Pembayaran\_Ikan (Pem\_IkanID, TanggalPembayaranIkan)

*Primary Key* Pem\_IkanID

1. Pemesanan (PemesananID, TanggalPemesanan)

*Primary Key* PemesananID

c. Identifikasi Tipe Relasi *Biner One-to-Many*(1:\*)

* Relasi antara Pabrik dan Bagian\_Pembayaran menyertakan *primary key* PabrikID dari entitas Pabrik sebagai *foreign key* pada Bagian\_Pembayaran.

Tabel Relasi antara Pabrik dan Bagian\_Pembayaran

|  |  |
| --- | --- |
| Pabrik (PabrikID, NamaPabrik, AlamatPabrik, TelpNumPabrik)  *Primary Key* PabrikID | Bagian\_Pembayaran (Bag\_PembayaranID, **PabrikID,** NamaStaff, Jabatan, Alamat, Gaji)  *Primary Key* Bag\_PembayaranID  *Foreign Key* **PabrikID** *References* Pabrik(PabrikID) |

* Relasi antara Bagian\_Pembayaran dan Pembayaran\_Ikan menyertakan *primary key* Bag\_PembayaranID dari entitas Bagian\_Pembayaran sebagai *foreign key* pada Pembayaran\_Ikan.

Tabel Relasi antara Bagian\_Pembayaran dan Pembayaran\_Ikan

|  |  |
| --- | --- |
| Bagian\_Pembayaran (Bag\_PembayaranID, NamaStaff, Jabatan, Alamat, Gaji)  *Primary Key* Bag\_PembayaranID | Pembayaran\_Ikan (Pem\_IkanID , **Bag\_PembayaranID,** TanggalPembayaranIkan)  *Primary Key* Pem\_IkanID  *Foreign Key* **Bag\_PembayaranID** *References* Bagian\_Pembayaran (**Bag\_PembayaranID**) |

* Relasi antara Nelayan dan Setor menyertakan *primary key* NelayanID dari entitas Nelayan sebagai *foreign key* pada Setor.

Tabel Relasi antara Nelayan dan Setor

|  |  |
| --- | --- |
| Nelayan (NelayanID, NamaNelayan, AlamatNelayan, DOB, Gender, NelayanTelpNum)  *Primary Key* NelayanID | Setor (SetorID, **NelayanID,** TanggalSetor)  *Primary Key* SetorID  *Foreign Key* **NelayanID** *References* Nelayan (NelayanID) |

* Relasi antara Bagian\_Pemesanan dan Pemesanan menyertakan *primary key* Bag\_PemesananID dari entitas Bagian\_Pemesanan sebagai *foreign key* pada Pemesanan.

Tabel Relasi antara Bagian\_Pemesanan dan Pemesanan

|  |  |
| --- | --- |
| Bagian\_Pemesanan (Bag\_PemesananID, NamaStaff, Jabatan, Alamat, Gaji)  *Primary Key* Bag\_PemesananID | Pemesanan (PemesananID, **Bag\_PemesananID,** TanggalPemesanan)  *Primary Key* PemesananID  *Foreign Key* **Bag\_PemesananID** *References* Bagian\_Pemesanan (Bag\_PemesananID) |

* Relasi antara Pelanggan dan Pemesanan menyertakan *primary key* PelangganID dari entitas Pelanggan sebagai *foreign key* pada Pemesanan.

Tabel Relasi antara Pelanggan dan Pemesanan

|  |  |
| --- | --- |
| Pelanggan (PelangganID, NamaPelanggan, AlamatPelanggan, DOB, Gender, PelangganTelpNum)  *Primary Key* PelangganID | Pemesanan (PemesananID, **PelangganID ,** TanggalPemesanan)  *Primary Key* PemesananID  *Foreign Key* **PelangganID** *References* Pelanggan(PelangganID) |

* Relasi antara Kurir dan Pengiriman menyertakan *primary key* KurirID dari entitas Kurir sebagai *foreign key* pada Pengiriman .

Tabel Relasi antara Kurir dan Pengiriman

|  |  |
| --- | --- |
| Kurir (KurirID, NamaKurir)  *Primary Key* KurirID | Pengiriman (PengirimanID, **KurirID,** TanggalPengiriman)  *Primary Key* PengirimanID  *Foreign Key* **KurirID** *References* Kurir (KurirID) |

* Relasi antara Pembayaran dan Penagihan menyertakan *primary key* PembayaranID dari entitas Pembayaran sebagai *foreign key* pada Penagihan .

Tabel Relasi antara Pembayaran dan Penagihan

|  |  |
| --- | --- |
| Pembayaran (PembayaranID, TanggalPembayaran)  *Primary Key* PembayaranID | Penagihan (PenagihanID, **PembayaranID,** TanggalPenagihan)  *Primary Key* PenagihanID  *Foreign Key* **PembayaranID** *References* Pembayaran (PembayaranID) |

d. Identifikasi Tipe *Relasi Biner One-to-One*(1:1)

* Relasi antara Setor dan Pembayaran\_Ikan menyertakan *primary key* SetorID dari entitas Setor sebagai *foreign key* pada Pembayaran\_Ikan.

Tabel Relasi antara Setor dan Pembayaran\_Ikan

|  |  |
| --- | --- |
| Setor (**SetorID**, TanggalSetor)  *Primary Key* SetorID | Pembayaran\_Ikan (Pem\_IkanID, **SetorID**, TanggalPembayaranIkan)  *Primary Key* Pem\_IkanID  *Foreign Key* SetorID *References* Setor (SetorID) |

* Relasi antara Pembayaran dan Pemesanan menyertakan *primary key* PembayaranID dari entitas Pembayaran sebagai *foreign key* pada Pemesanan.

Tabel Relasi antara Pemesanan dan Pembayaran

|  |  |
| --- | --- |
| Pemesanan (PemesananID, TanggalPemesanan)  *Primary Key* PemesananID | Pembayaran (**PemesananID**, PembayaranID, TanggalPembayaran)  *Primary Key* PembayaranID  *Foreign Key* PemesananID *References* Pemesanan (PemesananID) |

* Relasi antara Pembayaran dan Pengiriman menyertakan *primary key* PembayaranID dari entitas Pembayaran sebagai *foreign key* pada Pengiriman.

Tabel Relasi antara Pembayaran dan Pengiriman

|  |  |
| --- | --- |
| Pembayaran (**PembayaranID**, TanggalPembayaran)  *Primary Key* PembayaranID | Pengiriman (PengirimanID, **PembayaranID**, TanggalPengiriman)  *Primary Key* PengirimanID  *Foreign Key* PembayaranID References Pembayaran (PembayaranID) |

* Relasi antara Pengiriman dan Retur menyertakan *primary key* PengirimanID dari entitas Pengiriman sebagai *foreign key* pada Retur.

Tabel Relasi antara Pengiriman dan Retur

|  |  |
| --- | --- |
| Pengiriman (**PengirimanID**, TanggalPengiriman)  *Primary Key* PengirimanID | Retur (ReturID, **PengirimanID**, TanggalPengembalian, Alasan)  Primary Key ReturID  *Foreign Key* PengirimanID *References* Pengiriman (PengirimanID) |

e. Tipe Relasi *recursive One-To-One* (1..1)

* Tidak terdapat relasi rekursif.

f. Tipe Relasi *superclass* atau *subclass*

* Tidak terdapat relasi *superclass* atau *subclass*.

g. Tipe Relasi *Binary Many-To-Many* (\*..\*)

* Relasi antara tabel Setor dan Ikan muncul tabel DetailSetor (**SetorID, IkanID**). SetorID dan IkanID menjadi *primary key* dan *foreign key* di tabel DetailSetor.
* Relasi antara tabel Pemesanan dan Ikan muncul tabel DetailPemesanan (PemesananID, IkanID). PemesananID dan IkanID menjadi *primary key* dan *foreign key* di tabel DetailPemesanan.

h. Tipe Relasi Kompleks

* Tidak terdapat relasi yang kompleks, semua relasi adalah *binary.*

i. *Multi-valued attributes*

* Menghilangkan *Multi-Values Atributes*.

1. PelangganTelpNum

Atribut PelangganTelpNum yang didapat dari entitas Pelanggan dihilangkan dan menghasilkan entitas baru yang bernama TelpNumPelanggan.

Tabel Relasi antara Pelanggan dan TelpNumPelanggan

|  |  |
| --- | --- |
| Pelanggan(**PelangganID**,...)  *Primary Key* PelangganID | TelpNumPelanggan (PelangganID**,** **PelangganTelpNum**)  *Primary Key* PelangganTelpNum  *Foreign Key* PelangganID References Pelanggan (PelangganID) |

**2. Memvalidasi Relasi Menggunakan Normalisasi**

2.1 Validasi Setor ikan

1NF

Setor(**SetorID**,TanggalSetor,NelayanID,NamaNelayan,AlamatNelayan,DOB,Gender,NelayanTelpNum,{**IkanID**,NamaIkan, PurchasePrice,Weight})

2NF

Setor(**SetorID**,TanggalSetor,NelayanID,NamaNelayan,AlamatNelayan,DOB,Gender,NelayanTelpNum)

DetailSetor(**SetorID**, **IkanID,** Weight)

Ikan **(IkanID**,NamaIkan, PurchasePrice,Weight)

3NF

Setor(**SetorID**,TanggalSetor, NelayanID)

DetailSetor(**SetorID**, **IkanID,** Weight)

Ikan **(IkanID**,NamaIkan, PurchasePrice,Weight)

Nelayan(**NelayanID**,NamaNelayan,AlamatNelayan,DOB,Gender,NelayanTelpNum)

2.2 Validasi Pemesanan ikan

1NF

Pemesanan(**PemesananID**,TanggalPemesanan,PelangganID,NamaPelanggan,AlamatPelanggan,DOB,Gender,PelangganTelpNum,{**IkanID**,NamaIkan,SellPrice,Weight},Bag\_pemesananID,NamaStaff,Jabatan,Alamat,Gaji)

2NF

Pemesanan(**PemesananID**,TanggalPemesanan,PelangganID,NamaPelanggan,AlamatPelanggan,DOB,Gender,PelangganTelpNum, Bag\_pemesananID,NamaStaff,Jabatan,Alamat,Gaji)

DetailPemesanan(**PemesananID, IkanID,** Weight)

Ikan(**IkanID**,NamaIkan,SellPrice)

3NF

Pemesanan(**PemesananID**,TanggalPemesanan,PelangganID, Bag\_pemesananID)

DetailPemesanan(**PemesananID, IkanID,** Weight)

Ikan(**IkanID**,NamaIkan,SellPrice)

Pelanggan(**PelangganID**,NamaPelanggan,AlamatPelanggan,DOB,Gender)

TelpNumPelanggan(**PelangganTelpNum,** PelangganID)

Bagian\_pemesanan(**Bag\_pemesananID**,NamaStaff,Jabatan,Alamat,Gaji)

2.3 Validasi Pembayaran Ikan

1NF

Pembayaran\_Ikan(**Pem\_IkanID**,TanggalPembayaranIkan,SetorID,TanggalSetor,NelayanID,Bag\_pembayaranID,NamaStaff,Jabatan,Alamat,Gaji)

2NF

Tidak ada *partial depedency*

3NF

Pembayaran\_Ikan(**Pem\_IkanID**,TanggalPembayaranIkan, SetorID, Bag\_pembayaranID)

Bagian\_Pembayaran(**Bag\_pembayaranID**,NamaStaff,Jabatan,Alamat,Gaji,PabrikID)

Setor(**SetorID**,TanggalSetor,NelayanID)

Pabrik(**PabrikID**,NamaPabrik, AlamatPabrik,TelpNumPabrik)

2.4 Validasi Pembayaran Pemesanan Ikan

1NF

Pembayaran(**PembayaranID**,TanggalPembayaran,PemesananID,TanggalPemesanan,PelangganID,Bag\_pemesananID)

2NF

Tidak ada *partial depedency*

3NF

Pembayaran(**PembayaranID**,TanggalPembayaran, PemesananID)

Pemesanan(**PemesananID**,TanggalPemesanan,PelangganID,Bag\_pemesananID)

2.5 Validasi PenagihanPembayaran

1NF

Penagihan(**PenagihanID**,TanggalPenagihan,PembayaranID,TanggalPembayaran,PemesananID)

2NF

Tidak ada *partial depedency*

3NF

Penagihan(**PenagihanID**,TanggalPenagihan, PembayaranID)

Pembayaran(**PembayaranID**,TanggalPembayaran,PemesananID)

2.6 Validasi PengirimanI Ikan

1NF

Pengiriman(**PengirimanID**,TanggalPengiriman,PembayaranID, TanggalPembayaran,PemesananID,KurirID,NamaKurir)

2NF

Tidak ada *partial depedency*

3NF

Pengiriman(**PengirimanID**,TanggalPengiriman, PembayaranID, KurirID)

Pembayaran(**PembayaranID**, TanggalPembayaran,PemesananID)

Kurir(**KurirID**,NamaKurir)

2.7 Validasi Retur Ikan

1NF

Retur(**ReturID**,TanggalPengembalian,Alasan,PengirimanID,KurirID,PembayaranID)

2NF

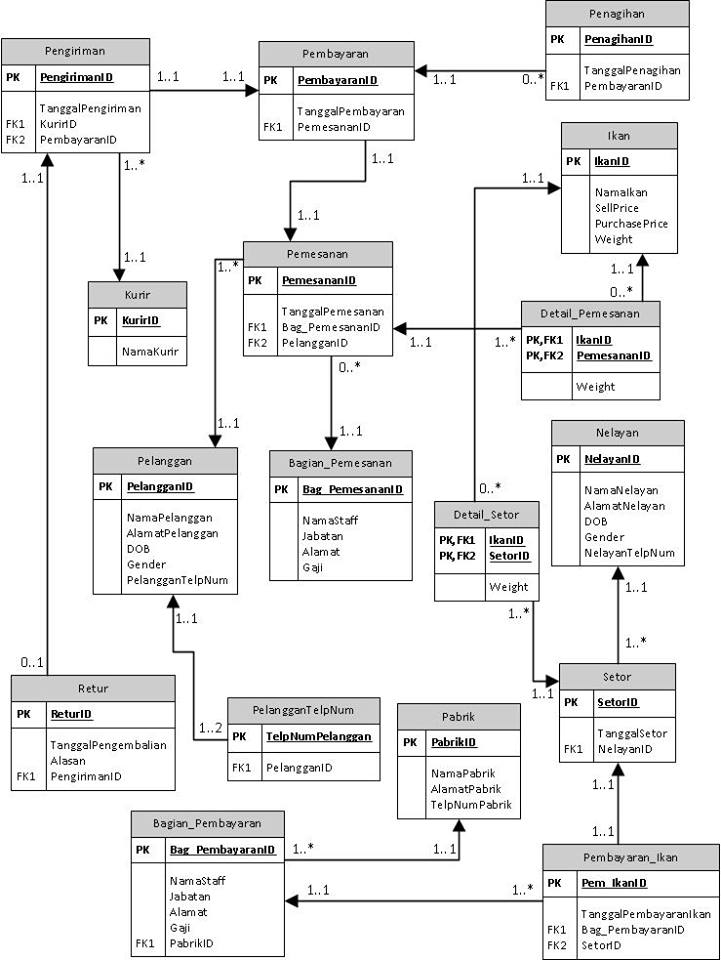
Tidak ada *partial depedency*

3NF

Retur(**ReturID**,TanggalPengembalian,Alasan, PengirimanID)

Pengiriman(**PengirimanID**,KurirID,PembayaranID)

**3. Memvalidasi Relasi terhadap transaksi *user***



**4. Memeriksa *Integrity Constraint***

Bertujuan untuk menetapkan *integrity constraint* yang diberikan dalam *view*.Adapun tipe-tipe *integrity constraint* yang ada adalah sebagai berikut:

* Data yang diperlukan (*Required data*)

Beberapa atribut harus selalu memiliki nilai valid (tidak boleh *NULL*).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nama Entitas | Atribut | Tipe Data | Length | Nulls | Multi Value |
| Pelanggan | PelangganID | Char | 9 | No | No |
| NamaPelanggan | Varchar | 50 | No | No |
| AlamatPelanggan | Varchar | 50 | No | No |
| DOB | Date | - | No | No |
| Gender | Char | 7 | No | No |
| PelangganTelpNum | Varchar | 13 | No | Yes |
|  | | | | |
| Ikan | IkanID | Char | 9 | No | No |
| NamaIkan | Varchar | 50 | No | No |
| SellPrice | Int | - | No | No |
| PurchasePrice | Int | - | No | No |
| Weight | Int | - | No | No |
| Nelayan | NelayanID | Char | 9 | No | No |
| NamaNelayan | Varchar | 50 | No | No |
| AlamatNelayan | Varchar | 50 | No | No |
| DOB | Date | - | No | No |
| Gender | Char | 7 | No | No |
| NelayanTelpNum | Varchar | 13 | No | No |
| Bagian\_Pembayaran | Bag\_PembayaranID | Char | 9 | No | No |
| NamaStaff | Varchar | 50 | No | No |
| Jabatan | Varchar | 30 | No | No |
| Alamat | Varchar | 50 | No | No |
| Gaji | Int |  | No | No |
| Pabrik | PabrikID | Char | 9 | No | No |
| NamaPabrik | Varchar | 50 | No | No |
| AlamatPabrik | Varchar | 50 | No | No |
| TelpNumPabrik | Varchar | 13 | No | No |
| Bagian\_  Pemesanan | Bag\_PemesananID | Char | 9 | No | No |
| NamaStaff | Varchar | 50 | No | No |
| Jabatan | Varchar | 30 | No | No |
| Alamat | Varchar | 50 | No | No |
| Gaji | Int |  | No | No |
| Kurir | KurirID | Char | 9 | No | No |
| NamaKurir | Varchar | 50 | No | No |
| Retur | ReturID | Char | 7 | No | No |
| TanggalPengembalian | Date |  | No | No |
| Alasan | Varchar | 100 | No | No |
| Pengiriman | PengirimanID | Char | 9 | No | No |
| TanggalPengiriman | Date |  | No | No |
| Setor | SetorID | Char | 9 | No | No |
| TanggalSetor | Date |  | No | No |
| Weight | Int |  | No | No |
| Pembayaran | PembayaranID | Char | 9 | No | No |
| TanggalPembayaran | Date |  | No | No |
| Penagihan | PenagihanID | Char | 9 | No | No |
| TanggalPenagihan | Date |  | No | No |
| Pembayaran\_  Ikan | Pem\_IkanID | Char | 9 | No | No |
| TanggalPembayaranIkan | Date |  | No | No |
| Pemesanan | PemesananID | Char | 9 | No | No |
| TanggalPemesanan | Date |  | No | No |
| Weight | Int | - | No | No |

Dalam kasus yang kami buat, tidak boleh ada atribut yang bernilai *NULL*, sehingga dapat dikatakan bahwa semua atribut pada setiap entitas adalah *required* atribut.

* Batasan atribut *domain* (*atribut domain constraint*)

*Domain integrity* dimaksudkan untuk menguji apakah atribut yang diuji telah sesuai dengan domain yang ditentukan atau tidak.

|  |  |  |
| --- | --- | --- |
| Nama Entitas | Atribut | Atribut Domain |
| Pelanggan | PelangganID | PELxxxx=karakter(0000-9999) |
| NamaPelanggan | Karakter A-Z,a-z |
| AlamatPelanggan | Karakter A-Z,a-z |
| DOB | dd/mm/YYYY |
| Gender | “Male” atau “Female” |
| PelangganTelpNum | 08xxxxxxxxxx |
| Ikan | IkanID | IKNxxxx=karakter(0000-9999) |
| NamaIkan | Karakter A-Z,a-z |
| SellPrice |  |
| PurchasePrice |  |
| Weight |  |
| Nelayan | NelayanID | NLYxxxx=karakter(0000-9999) |
| NamaNelayan | Karakter A-Z,a-z |
| AlamatNelayan | Karakter A-Z,a-z |
| DOB | Dd/mm/YYYY |
| Gender | “Male” atau “Female” |
| NelayanTelpNum | 08xxxxxxxxxx |
| Bagian\_Pembayaran | Bag\_PembayaranID | BPBxxxx=karakter(0000-9999) |
| NamaStaff | Karakter A-Z,a-z |
| Jabatan | Karakter A-Z,a-z |
| Alamat | Karakter A-Z,a-z |
| Gaji |  |
| Pabrik | PabrikID | PBKxxxx=karakter(0000-9999) |
| NamaPabrik | Tanggal (dd-mm-yyyy) |
| AlamatPabrik | Karakter A-Z,a-z |
| TelpNumPabrik | 08xxxxxxxxxx |
| Bagian\_Pemesanan | Bag\_PemesananID | BPMxxxx=karakter(0000-9999) |
| NamaStaff | Karakter A-Z,a-z |
| Jabatan | Karakter A-Z,a-z |
| Alamat | Karakter A-Z,a-z |
| Gaji |  |
| Kurir | KurirID | KRRxxxx=karakter(0000-9999) |
| NamaKurir | Karakter A-Z,a-z |
|  |  |
|  |
| Retur | ReturID | RTRxxxx=karakter(0000-9999) |
| TanggalPengembalian | dd/mm/YYYY |
| Alasan | Karakter A-Z,a-z |
| Pengiriman | PengirimanID | PGRxxxx=karakter(0000-9999) |
| TanggalPengiriman | dd/mm/YYYY |
| Setor | SetorID | STRxxxx=karakter(0000-9999) |
| TanggalSetor | dd/mm/YYYY |
| Weight |  |
| Pembayaran | PembayaranID | PBYxxxx=karakter(0000-9999) |
| TanggalPembayaran | dd/mm/YYYY |
| Penagihan | PenagihanID | PNGxxxx=karakter(0000-9999) |
| TanggalPenagihan | dd/mm/YYYY |
| Pembayaran Ikan | Pem\_IkanID | PBIxxxx=karakter(0000-9999) |
| TanggalPembayaranIkan | dd/mm/YYYY |
| Pemesanan | PemesananID | PMNxxxx=karakter(0000-9999) |
| TanggalPemesanan | dd/mm/YYYY |
| Weight |  |

* *Multiplicity*

*Multiplicity* merepresentasikan batasan-batasan atau *constraint* yang ada pada relasi antar suatu data dengan data lainnya yang ada di dalam *database*.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nama Entitas | Multiplicity | Relasi | Nama Entitas | Multiplicity |
| Pengiriman | 1..1  1..\* | Memiliki  Dilakukan | Pembayaran  Kurir | 1..1  1..1 |
| Pembayaran | 1...1 | Bagian dari | Pemesanan | 1...1 |
| Penagihan | 0..\* | Bagian dari | Pembayaran | 1..1 |
| Retur | 0..1 | Bagian dari | Pengiriman | 1...1 |
| Pemesanan | 1..\*  0..\* | Dilakukan  Ditangani | Pelanggan  Bagian­\_Pemesanan | 1..1  1..1 |
| Detail\_Pemesanan | 1..\*  0..\* | Mempunyai  Bagian dari | Ikan  Pemesanan | 1..1  1..1 |
| Detail\_Setor | 0..\*  1..\* | Mempunyai  Bagian dari | Ikan  Setor | 1..1  1..1 |
| Setor | 1..\* | Dilakukan | Nelayan | 1..1 |
| Gallery | 0...\* | Dibuat | Member | 1…1 |
| Bagian\_Pembayaran | 1..\* | Membayar ke | Pabrik | 1..1 |
| Pembayaran\_Ikan | 1..\* | Ditangani | Bagian\_Pembayaran | 1..1 |

* *Entity integrity*

*Primary key* dari suatu entitas tidak boleh bernilai *NULL*. Dalam hal ini juga ditentukan *candidate key* serta *alternate key* dari suatu entitas.

|  |  |  |  |
| --- | --- | --- | --- |
| Nama Entitas | Candidate Key | Primary Key | Alternate Key |
| Pelanggan | PelangganID | PelangganID | NamaPelanggan |
| NamaPelanggan |
| AlamatPelanggan |
| DOB |
| Gender |
| PelangganTelpNum |
| Ikan | IkanID | IkanID | NamaIkan |
| NamaIkan |
| SellPrice |
| PurchasePrice |
| Weight |
| Nelayan | NelayanID | NelayanID | NamaNelayan |
| NamaNelayan |
| AlamatNelayan |
| DOB |
| Gender |
| NelayanTelpNum |
| Bagian\_Pembayaran | Bag\_PembayaranID | Bag\_PembayaranID | NamaStaff |
| NamaStaff |  |
| Jabatan |
| Alamat |
| Gaji |
| Pabrik | PabrikID | PabrikID | NamaPabrik |
| NamaPabrik |
| AlamatPabrik |
| TelpNumPabrik |
| Bagian\_Pemesanan | Bag\_PemesananID | Bag\_PemesananID | NamaStaff |
| NamaStaff |  |
| Jabatan |
| Alamat |
| Gaji |
| Kurir | KurirID | KurirID | NamaKurir |
| NamaKurir |
| Retur | ReturID | ReturID | TanggalPengembalian |
| TanggalPengembalian |
| Alasan |
| Pengiriman | PengirimanID | PengirimanID | TanggalPengiriman |
| TanggalPengiriman |
| Setor | SetorID | SetorID | TanggalSetor |
| TanggalSetor |
| Weight |
| Pembayaran | PembayaranID | PembayaranID | TanggalPembayaran |
| TanggalPembayaran |
| Penagihan | PenagihanID | PenagihanID | TanggalPenagihan |
| TanggalPenagihan |
| Pembayaran\_Ikan | Pem\_IkanID | Pem\_IkanID | TanggalPembayaranIkan |
| TanggalPembayaranIkan |
| Pemesanan | PemesananID | PemesananID | TanggalPemesanan |
| TanggalPemesanan |
| Weight |

• *Referential Integrity*

*Referential integrity* dimaksudkan untuk menguji apakah setiap tabel yang diuji telah dapat merujuk ke tabel lain yang saling berhubungan.

|  |
| --- |
| Bagian\_Pembayaran(Bag\_PembayaranID, NamaStaff, Jabatan, Alamat, Gaji,PabrikID)  *Primary Key* Bag\_PembayaranID  *Foreign Key* PabrikID *References* Pabrik(PabrikID) *ON UPDATE CASCADE ON DELETE CASCADE* |
| Pembayaran\_Ikan (Pem\_IkanID, TanggalPembayaranIkan,Bag\_PembayaranID,SetorID)  *Primary Key* Pem\_IkanID  *Foreign Key* Bag\_PembayaranID *References* Bagian\_Pembayaran(Bag\_PembayaranID) *ON UPDATE CASCADE ON DELETE CASCADE*  *Foreign Key* SetorID *References* Setor(SetorID) *ON UPDATE CASCADE ON DELETE CASCADE* |
| Setor (SetorID, TanggalSetor,NelayanID)  *Primary Key* SetorID  *Foreign Key* NelayanID *References* Nelayan(NelayanID) *ON UPDATE CASCADE ON DELETE CASCADE* |
| DetailSetor(SetorID, IkanID, Weight)  *Primary Key* SetorID,IkanID  *Foreign Key* SetorID *References* Setor(SetorID) *ON UPDATE CASCADE ON DELETE CASCADE*  *Foreign Key* IkanID *References* Ikan(IkanID) *ON UPDATE CASCADE ON DELETE CASCADE* |
| Pemesanan(PemesananID,TanggalPemesanan,PelangganID, Bag\_pemesananID)  *Primary Key* PemesananID  *Foreign Key* PelangganID *References* Pelanggan(PelangganID) *ON UPDATE CASCADE ON DELETE CASCADE*  *Foreign Key* Bag\_pemesananID *References* Bagian\_pemesanan(Bag\_pemesananID) *ON UPDATE CASCADE ON DELETE CASCADE* |
| DetailPemesanan(PemesananID**,** IkanID**,** Weight)  *Primary Key* PemesananID, IkanID  *Foreign Key* PemesananID *References* Pemesanan(PemesananID) *ON UPDATE CASCADE ON DELETE CASCADE*  *Foreign Key* IkanID *References* Ikan(IkanID) *ON UPDATE CASCADE ON DELETE CASCADE* |
| TelpNumPelanggan(PelangganTelpNum,PelangganID)  *Primary Key* PelangganTelpNum  *Foreign Key* PelangganID *References* Pelanggan (PelangganID) *ON UPDATE CASCADE ON DELETE CASCADE* |
| Pembayaran\_Ikan(Pem\_IkanID,TanggalPembayaranIkan, SetorID, Bag\_pembayaranID)  *Primary Key* Pem\_IkanID  *Foreign Key* SetorID *References* Setor(SetorID) *ON UPDATE CASCADE ON DELETE CASCADE*  *Foreign Key* Bag\_pembayaranID *References* Bagian\_pembayaran(Bag\_pembayaranID) *ON* *UPDATE CASCADE ON DELETE CASCADE* |
| Pembayaran(PembayaranID,TanggalPembayaran, PemesananID)  *Primary Key* PembayaranID  *Foreign Key* PemesananID *References* Pemesanan(PemesananID) *ON UPDATE CASCADE ON DELETE CASCADE* |
| Penagihan(PenagihanID,TanggalPenagihan, PembayaranID)  *Primary Key* PenagihanID  *Foreign Key* PembayaranID *References* Pembayaran(PembayaranID) *ON UPDATE CASCADE ON DELETE CASCADE* |
| Pengiriman(PengirimanID,TanggalPengiriman, PembayaranID, KurirID)  *Primary Key* PengirimanID  *Foreign Key* PembayaranID *References* Pembayaran(PembayaranID) *ON UPDATE CASCADE ON DELETE CASCADE*  *Foreign Key* KurirID *References* Kurir(KurirID) *ON UPDATE CASCADE ON DELETE CASCADE* |
| Retur(ReturID,TanggalPengembalian,Alasan, PengirimanID)  *Primary Key* ReturID  *Foreign Key* PengirimanID *References* Pengiriman(PengirimanID) *ON UPDATE CASCADE ON DELETE CASCADE* |
|  |

Jika data di-*update* atau dihapus, maka data *foreign key* pada entitas lainnya akan ikut ter-*update* atau terhapus. (*ON UPDATE CASCADE ON DELETE CASCADE*)

* *General Constraint*

Pada *system* yang akan kami buat, kami belum memiliki batasan atau *constraint* umum yang membatasi suatu transaksi.

**5. Melakukan *Review Model Data* Logikal Dengan *User***

Tujuan dari langkah ini adalah untuk memvalidasi logikal model data sehingga dapat mendukung transaksi yang diperlukan sesuai dengan spesifikasi kebutuhan perusahaan atau pengguna. Jika pengguna tidak puas dengan model, maka beberapa pengulangan langkah sebelumnya dalam metodolgi mungkin diperlukan. Cara-cara yang diperlukan untuk melakukan *review model data* logikal dengan *user* dapat dilakukan dengan cara menghubungkan antara model data logis dan *data flow diagram* dimana :

* Setiap *datastore* perusahaan atau pengguna harus mewakili seluruh jumlah jenis entitas.
* Atribut dalam setiap arus data harus milik tipe entitas.

Yang perlu diperhatikan tentang *data store* :

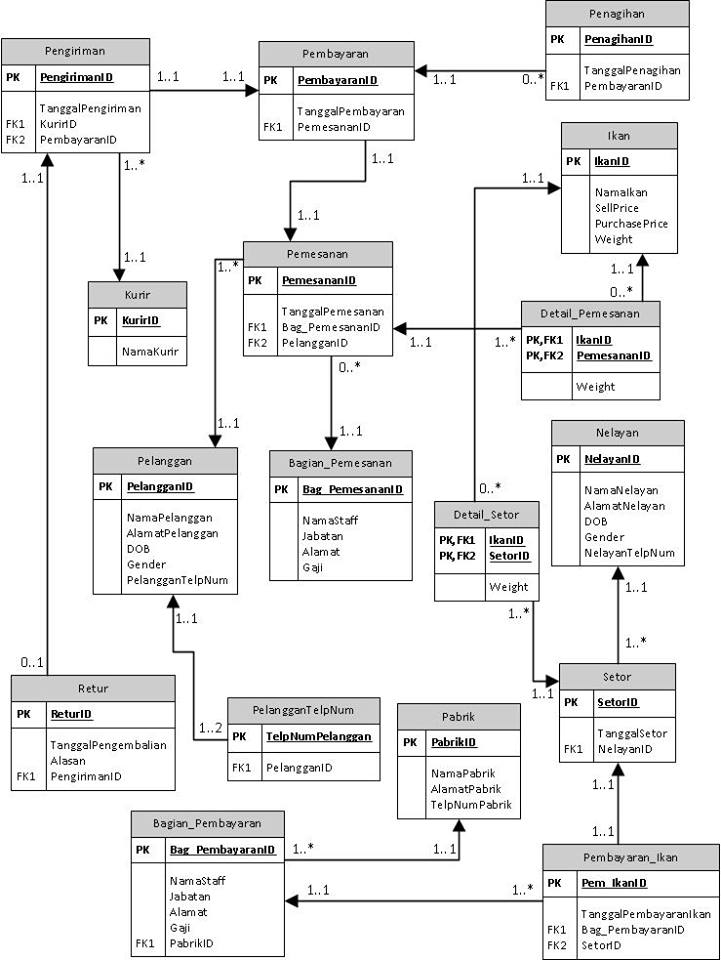
1. Alur data dari proses menuju *data store*, hal ini berarti *data store* berfungsi sebagai tujuan atau tempat penyimpanan dari suatu proses (proses *write*).

2. Alur data dari *data store* ke proses, hal ini berarti *data store* berfungsi sebagai sumber atau proses memerlukan data (proses *read*).

3. Alur data dari proses menuju *data store* dan sebaliknya berarti berfungsi sebagai sumber dan tujuan.

Untuk memenuhi hal ini, perlu dilakukan diskusi dengan perwakilan perusahaan atau pengguna. Setelah diskusi tersebut, model data logikal ini dapat dinyatakan telah memenuhi kriteria-kriteria yang diperlukan oleh perusahaan atau pengguna.

**6. Menggabungkan *Model Data* Logikal kedalam *Model Global***



**7. Cek untuk Pertumbuhan di Masa Mendatang**

Tujuan dari langkah ini adalah untuk menentukan apakah akan ada terjadi perubahan yang signifikan di masa yang akan datang, dan apakah *model data* logikal sudah bisa mengakomodasi perubahan tersebut. Tentu saja hal ini dibutuhkan diskusi dengan perwakilan perusahaan atau pengguna. Melalui diskusi dengan pihak perusahaan atau pengguna, *model data* logikal yang telah dibuat ini dapat mengakomodasi perubahan-perubahan yang mungkin terjadi dimasa yang akan datang.