PERANCANGAN DATABASE SISTEM INFORMASI AKUNTANSI MENGGUNAKAN KOMBINASI REA MODEL, ERD, DAN NORMALISASI DATA

Michael Iskandar, Christian C. Henry, Asdi Aulia
Fakultas Ekonomi, Universitas Katolik Parahyangan

Abstract

In the field of accounting, information technology becomes the standard tool of the trade. It allows for much more accurate and faster processing of accounting data, while keeping costs down. These advantages require accountants to understand the basics of information technology related to information processing. One of the most important is the knowledge about database design, in which comprehension will ensure that all necessary data for financial statements are captured and business rules are adhered to. This paper discusses a database design method which combines the data normalization process, entity-relationship approach and resources-events-agents model.

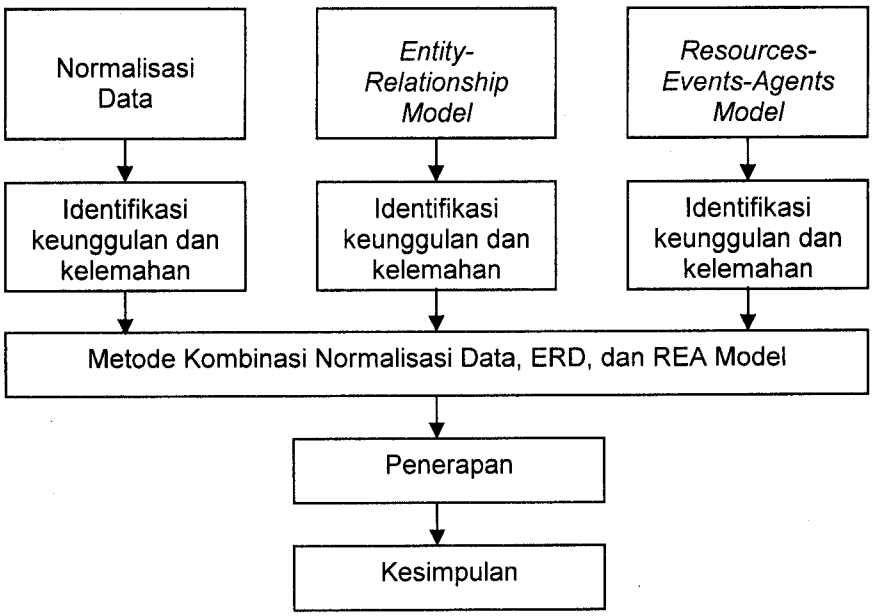
Key words: *database, data normalization, entity-relationship diagram, resources-events-agents model.*

Pendahuluan

Bersama dengan semakin pentingnya peran komputer dalam sistem informasi akuntansi, maka peran *database* sebagai tempat penyimpanan data akuntansi menjadi semakin penting pula. Pengetahuan tentang perancangan *database* yang baik menjadi pengetahuan yang harus dikuasai oleh akuntan maupun calon akuntan, karena dalam menjalani profesinya dia harus dapat memberi masukan pada proyek pengembangan sistem informasi akuntansi perusahaan, serta harus dapat mengkaji sebuah *database* perusahaan ketika melakukan kegiatan *audit*.

Saat ini telah terdapat sejumlah proses, pendekatan, atau model (yang tidak bersifat *mutually exclusive*) untuk merancang *relational database*. Tiga di antaranya adalah: proses normalisasi data yang diperkenalkan Codd (1970), pendekatan *Entity-Relationship Model* (beserta perkakasnya, *Entity-Relationship Diagram* atau ERD) yang diciptakan oleh Chen (1976), serta *Resources-Events-Agents Model* (REA Model) yang ditemukan oleh McCarthy (1982). Masing-masing proses, pendekatan, atau model memiliki keunggulan dan kelemahannya sendiri-sendiri. Di dalam makalah ini dijelaskan sebuah metode perancangan *database* yang mengkombinasikan kekuatan dari masing-masing proses, pendekatan, atau model tersebut, dengan harapan hal ini dapat menghilangkan kelemahan-kelemahan yang ada.

Gambar 1 menunjukkan kerangka pemikiran dari makalah ini. Pertama-tama dijelaskan tentang masing-masing proses, pendekatan, atau model secara terpisah. Kemudian, diidentifikasi keunggulan dan kelemahan masing-masing terkait dengan perancangan *database*, khususnya untuk sistem informasi akuntansi. Berdasarkan hasil identifikasi tersebut disusunlah sebuah metode kombinasi, yang kemudian dibuktikan efektivitasnya dengan cara menerapkannya atas sebuah kasus (perusahaan). Akhirnya, berdasarkan hasil penerapan tadi ditarik kesimpulan.



Gambar 1. Kerangka Pemikiran

Normalisasi Data

Database adalah sekumpulan *data files* yang saling berhubungan dan dikendalikan secara terpusat yang menyimpan data dengan sesedikit mungkin *data redundancy*. *Database* mengkonsolidasikan banyak *records* yang dahulu disimpan di dalam *files* yang terpisah-pisah ke dalam satu penampung *records,* sehingga bisa digunakan oleh berbagai *users* dan aplikasi pengolah data (Romney dan Steinbart, 2009: 805). *Database* bisa diorganisasi dengan beberapa cara, namun yang dibahas di sini adalah cara yang paling umum digunakan, yaitu *relational database*.

Menurut Romney, terdapat dua cara untuk merancang *relational database* yang terstruktur dengan baik. Cara pertama adalah normalisasi (data) dan cara lainnya adalah *semantic data modeling* (Romney dan Steinbart, 2009: 139) .

Normalisasi data berangkat dari sebuah tabel yang berisi banyak data dan kompleks. Tabel ini kemudian diorganisasi menjadi beberapa tabel yang lebih kecil dan memiliki struktur data yang lebih stabil. Terdapat tiga langkah di dalam normalisasi data, yaitu (Kendall dan Kendall, 2005: 456-457):

1. Menghilangkan kelompok berulang/*repeating groups*.
2. Menghilangkan ketergantungan parsial/*partial dependencies*.
3. Menghilangkan ketergantungan transitif/*transitive dependencies*.

Berikut adalah contoh penerapan ketiga langkah normalisasi data. Misalkan tabel awal yang akan dinormalisasi adalah seperti yang ditampilkan dalam Tabel 1.

Tabel 1. Bentuk Tak Ternormalisasi

No Order	Kode Buku	Judul	Pengarang	Kode Supplier	Nama Supplier	Kuantitas Pesan
PO1	AIS-B01	Accounting Information Systems	Bodnar, Hopwood	S123	PT. Asal	25
PO1	AIS-H01	Accounting Information Systems	Hall	S123	PT. Asal	10
PO2	AIS-R01	Accounting Information Systems	Romney, Steinbart	S234	PT. Sembarang	15
PO2	AIS-W01	Accounting Information Systems	Wilkinson, Cerullo, Raval, Wong-on-Wing	S234	PT. Sembarang	5

Keterangan: Kode Buku tertentu hanya dipasok oleh *supplier* tertentu. Misal: PT. Asal hanya memasok buku AIS-B01 dan AIS-H01 dan tidak memasok AIS-R01 dan AIS-W01.

Langkah pertama adalah menghilangkan kelompok berulang (*repeating groups*). *Repeating groups* adalah penempatan beberapa fakta pada satu baris yang sama. Pada Tabel 1 terlihat bahwa pada baris pertama tersimpan beberapa fakta pada kolom "Pengarang", yaitu terdapat Bodnar dan Hopwood. Demikian pula dengan baris kedua, ketiga, dan keempat. Hal ini harus dihilangkan. Jadi, tabel pertama ini dipecah sehingga menjadi seperti yang tampak dalam Tabel 2A dan Tabel 2B.

Tabel 2A. Bentuk 1NF (A)

Kode Buku	Pengarang
AIS-B01	Bodnar
AIS-B01	Hopwood
AIS-H01	Hall
AIS-R01	Romney
AIS-R01	Steinbart
AIS-W01	Wilkinson
AIS-W01	Cerullo
AIS-W01	Raval
AIS-W01	Wong-on-Wing

Tabel 2B. Bentuk 1NF (B)

No Order	Kode Buku	Judul	Kode Supplier	Nama Supplier	Kuantitas Pesan
PO1	AIS-B01	Accounting Information Systems	S123	PT. Asal	25
PO1	AIS-H01	Accounting Information Systems	S123	PT. Asal	10
PO2	AIS-R01	Accounting Information Systems	S234	PT. Sembarang	15
PO2	AIS-W01	Accounting Information Systems	S234	PT. Sembarang	5

Kedua tabel tersebut (Tabel 2A dan Tabel 2B) sudah bebas dari *repeating groups,* dan disebut dalam kondisi *First Normal Form* (1NF).

Langkah kedua adalah menghilangkan ketergantungan parsial (*partial dependencies*). Ketergantungan parsial adalah bahwa ada *field* yang hanya diidentifikasi (bergantung pada) salah satu atau sebagian *primary key field* saja. *Primary key* pada Tabel 2B adalah "No. Order - Kode Buku.". *Primary key* yang terdiri dari lebih dari satu *field* seperti ini disebut *concatenated primary key* (Dunn, et al., 2005:54). *Field "Judul", "Kode Supplier"* dan *"Nama Supplier"* hanya bergantung pada *field "Kode Buku"*, bukan pada *concatenated primary key* dari Tabel 2B ini. Ketergantungan parsial seperti ini harus dihilangkan. Dengan demikian Tabel 2B dipecah menjadi Tabel 3A dan Tabel 3B.

Tabel 3A. Bentuk 2NF (A)

No Order	Kode Buku	Kuantitas Pesan
PO1	AIS-B01	25
PO1	AIS-H01	10
PO2	AIS-R01	15
PO2	AIS-W01	5

Tabel 3B. Bentuk 2NF (B)

Kode Buku	Judul	Kode Supplier	Nama Supplier
AIS-B01	Accounting Information Systems	S123	PT. Asal
AIS-H01	Accounting Information Systems	S123	PT. Asal
AIS-R01	Accounting Information Systems	S234	PT. Sembarang
AIS-W01	Accounting Information Systems	S234	PT. Sembarang

Ketiga tabel tersebut (Tabel 2A, 3A, dan 3B) sudah bebas dari *partial dependencies,* dan disebut dalam kondisi *Second Normal Form* (2NF).

Langkah ketiga adalah menghilangkan ketergantungan transitif (*transitive dependencies*). Ketergantungan transitif adalah adanya *non-primary key field* yang diidentifikasi oleh *non-primary key field* lain. *Primary key* pada Tabel 3B adalah *field "Kode Buku"*.

Non-primary key field "Nama Supplier" diidentifikasi oleh *non-primary key field "Kode Supplier"*. Ketergantungan transitif seperti ini harus dihilangkan. Dengan demikian Tabel 3B dipecah menjadi Tabel 4A dan Tabel 4B.

Tabel 4A. Bentuk 3NF (A)

Kode Buku	Judul	Kode Supplier
AIS-B01	Accounting Information Systems	S123
AIS-H01	Accounting Information Systems	S123
AIS-R01	Accounting Information Systems	S234
AIS-W01	Accounting Information Systems	S234

Tabel 4B. Bentuk 3NF (B)

Kode Supplier	Nama Supplier
S123	PT. Asal
S234	PT. Sembarang

Keempat tabel tersebut (Tabel 2A, 3A, 4A, dan 4B) sudah bebas dari *transitive dependencies,* dan disebut dalam kondisi *Third Normal Form* (3NF). Jadi, keempat tabel yang baru disebutkan ini adalah hasil akhir dari normalisasi data.

***Entity Relationship Diagram* (ERD)**

Entity-Relationship Model adalah salah satu dari model data. Pemodelan data (*data modeling*) adalah teknik untuk mengorganisasikan dan mendokumentasikan data dari suatu sistem (Whitten et al., 2008: 207). *Entity-Relationship Model* menggunakan *Entity-Relationship Diagram* (ERD) untuk mengorganisasikan dan mendokumentasikan data tersebut. Untuk memahami ERD, perlu dipahami terlebih dulu tentang konsep *entities, attributes, primary key, foreign key, relationships*, dan *cardinality.*

Entitas (*entity*) adalah sesuatu yang perlu disimpan datanya oleh perusahaan. Contoh dari entitas adalah pelanggan, departemen, ruang, mesin, pembelian, mata kuliah. Atribut (*attribute*) adalah karakteristik yang dimiliki oleh sebuah entitas atau sebuah hubungan (*relationship*). Sebagai contoh, untuk entitas "mahasiswa", atributnya adalah nama, alamat, nomor telepon, tanggal lahir, Indeks Prestasi Kumulatif (IPK), dan sebagainya. *Primary key* adalah atribut yang secara unik dan universal membedakan sebuah anggota entitas (*instance*) dengan anggota entitas lainnya. Sebagai contoh, untuk entitas "mahasiswa", yang membedakan seorang mahasiswa dengan mahasiswa lainnya adalah Nomor Pokok Mahasiswa (NPM). *Foreign key* adalah atribut dari sebuah tabel yang menjadi atribut di tabel lain yang fungsinya untuk menciptakan hubungan di antara kedua tabel tersebut. *Relationships* adalah asosiasi yang terdapat di antara satu entitas dengan entitas lainnya. *Cardinality* merepresentasikan aturan perusahaan tentang berapa kali sebuah anggota entitas boleh berhubungan dengan entitas lain (Dunn, et al., 2005: 52-55).

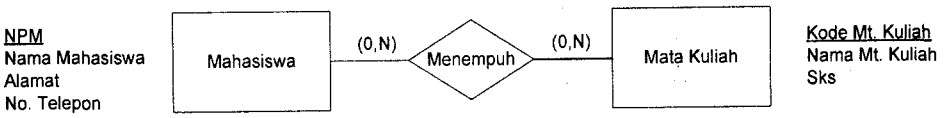
Tabel 5 menampilkan contoh untuk menjelaskan beberapa konsep tersebut.

Tabel 5. Tabel Mahasiswa

NPM	Nama Mahasiswa	Alamat	No. Telepon
2002110001	Albert Alvin	Jl. Aceh 1001 Bandung	7017010
2003120200	Budi Budiman	Jl. Bengkulu 2002 Bandung	7778888
2003130201	Charlie Chandra	Jl. Candi 3003 Bandung	7897890

Pada Tabel 5 terdapat *entity* mahasiswa. Mahasiswa memiliki atribut NPM (Nomor Pokok Mahasiswa), Nama Mahasiswa, Alamat, dan No. Telepon. *Primary key* dari Tabel Mahasiswa ini adalah NPM, karena nilai pada *field* NPM-lah yang secara unik dan universal membedakan sebuah anggota entitas (*instance*) dengan anggota entitas lainnya.

Notasi (metode merepresentasikan sesuatu dengan gambar atau tanda) untuk ERD ada beberapa macam, antara lain notasi Chen, Martin, Bachman, Merise, IDEFIX (Whitten et al., 2008: 208). Gambar 2 adalah contoh dari notasi Chen.



Gambar 2. Notasi ERD Menurut Chen

Pada Gambar 2 terdapat dua entitas, yaitu Mahasiswa dan Mata Kuliah. *Primary key* Mahasiswa adalah NPM dan *primary key* Mata Kuliah adalah Kode Mata Kuliah. Kedua entitas ini memiliki hubungan (*relationship*) bernama "Menempuh". Jadi, mahasiswa menempuh mata kuliah. Pada setiap entitas, dituliskan (*minimum cardinality*,*maximum cardinality*). *Cardinality* mahasiswa yang ditulis (0,N) dijelaskan sebagai berikut: *Minimum cardinality* 0 berarti bahwa seorang mahasiswa tidak wajib menempuh mata kuliah. (Seandainya *minimum cardinality* di sini bernilai 1, berarti seorang mahasiswa wajib menempuh mata kuliah.) *Maximum cardinality* N berarti bahwa seorang mahasiswa boleh menempuh lebih dari satu mata kuliah. (Seandainya *maximum cardinality* di sini bernilai 1, berarti seorang mahasiswa hanya boleh menempuh satu mata kuliah.)

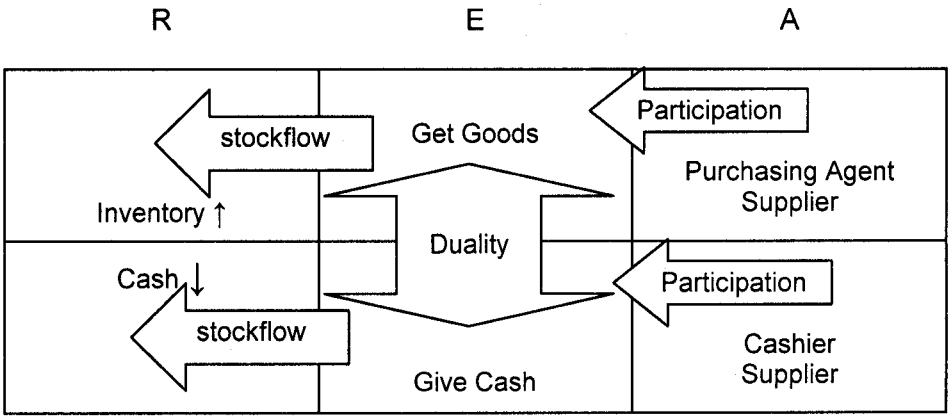
Hal yang sama berlaku juga untuk *cardinality* Mata Kuliah yang kebetulan sama, yaitu (0,N). *Minimum cardinality* 0 berarti bahwa sebuah mata kuliah tidak wajib ditempuh oleh mahasiswa. (Seandainya *minimum cardinality* di sini bernilai 1, berarti sebuah mata kuliah wajib ditempuh oleh mahasiswa.) *Maximum cardinality* N berarti bahwa sebuah mata kuliah boleh ditempuh oleh lebih dari satu mahasiswa. (Seandainya *minimum cardinality* di sini bernilai 1, berarti sebuah mata kuliah hanya boleh ditempuh oleh satu mahasiswa.)

Pemodelan REA (*Resources - Events -Agents*)

REA Model, atau secara lengkap disebut *Resources - Events - Agents Model*, pertama dikembangkan oleh McCarthy pada tahun 1982. Landasan dari pengembangan model ini adalah hasil observasi beliau atas berbagai sistem informasi akuntansi perusahaan-perusahaan (Dunn, et al., 2005:24).

Dunn, et al., (2005) menjelaskan adanya empat tingkat REA Model, yaitu: *value system level, value chain level, business process level,* dan *task level.* Perbedaan dari keempat tingkat itu adalah fokus pengamatannya. Pada *value system level*, digambarkan aliran sumber daya yang terjadi antara perusahaan dengan pihak-pihak luar perusahaan (Dunn, et al., 2005:40-42). Setelah arus sumber daya dari dan ke luar perusahaan telah teridentifikasi, maka diidentifikasi pula bagaimana arus sumber daya tadi di dalam perusahaan, khususnya antara siklus-siklus yang terdapat di dalam perusahaan: *revenue cycle, expenditure cycle, payroll cycle, production (manufacturing) cycle*, dan *financing cycle.* Model yang menggambarkan ini disebut dengan *Value Chain Level REA Model* (Dunn, et al., 2005:42-44; Romney and Steinbart, 2009:54). Setiap *business process REA Model* terfokus pada salah satu dari siklus-siklus yang telah disebut di atas, dan secara spesifik mengidentifikasi semua *resources, events*, dan *agents* yang terkait. Terakhir, *task level REA Model* dimaksudkan untuk menunjukkan prosedur-prosedur yang harus diselesaikan untuk mendukung proses bisnis yang telah digambar dalam *Business Process REA Model.* Pada prakteknya, *task level REA Model* tidak lain adalah *Data Flow Diagram* serta *Flowchart* (Dunn, et al., 2005:91-115).

Core REA Business Model selalu dibuat seputar konsep *duality* antara dua *events*, di mana *event* yang satu menyebabkan sebuah *resource* bertambah, tetapi *event* yang lain yang terkait menyebabkan *resource* lain berkurang. Aliran *resource* yang bertambah dan berkurang itu dikenal dengan istilah *stockflow.* Selain itu, juga terdapat *Agents* yaitu orang atau lembaga yang terkait dengan *event* yang bersangkutan. Pada umumnya terdapat minimal dua *agents*, yaitu *agent* yang menyerahkan *resource* tertentu, dan *agent* lain yang menerima *resource* itu (Dunn, et al., 2005:62-64). Meskipun umumnya *REA Business Process Model* digambar menggunakan ERD (lihat ilustrasi Dunn, et al., 2005:63; Romney dan Steinbart, 2009:591; serta Hall, 2008:498-500), dalam makalah ini model tersebut digambar menggunakan diagram sederhana yang disebut Diagram Kotak. Gambar 3 menunjukkan contoh diagram kotak itu untuk siklus pembelian (*expenditure cycle*).



Gambar 3. Diagram Kotak REA Model

Beranjak dari *Core REA Business Model*, dikembangkan pula *Extended REA Business Model,* di mana juga dikenal *instigation events, commitment events* dan *reversal events*. Secara singkat dapat dijelaskan bahwa *instigation event* adalah peristiwa yang mengawali sebuah aktivitas, *commitment event* adalah peristiwa di mana terbentuk suatu komitmen dari satu pihak ke pihak lain, dan *reversal event* adalah peristiwa di mana aliran sumber daya menjadi terbalik dari biasanya (Dunn, et al., 2005:82-86). Sebagai contoh, penerbitan sebuah *Purchase Requisition* merupakan hal yang mengawali aktivitas pembelian, maka dari itu *purchase requisition* merupakan *instigation event*. Sedangkan penerbitan suatu *Purchase Order* merupakan *commitment event*, karena dengan PO tersebut, perusahaan membentuk komitmen dengan pemasok. Jika setelah pembelian ternyata ada barang yang diretur ke pemasok, maka itu adalah *reversal event.*

REA Model dapat dipergunakan untuk merancang *database* secara langsung. Caranya adalah dengan menggambarkan setiap *resource*, *event*, dan *agent* sebagai entitas dalam ERD, serta hubungan-hubungan *duality, stockflow* dan *participation* sebagai relationship dalam ERD tersebut (Dunn, et al., 2005: 62-86).

Identifikasi Keunggulan Dan Kelemahan Tiga Cara Perancangan *Database*

Proses normalisasi data memiliki keunggulan berupa aturan-aturan yang jelas dan mudah untuk diikuti untuk memperoleh rancangan *relational database* yang baik. Namun demikian, seperti yang dapat dilihat pada contoh yang telah dibahas, normalisasi data berangkat dari satu tabel saja. Padahal, sebuah tabel dalam sebuah perusahaan belum tentu mencatat segala hal yang dibutuhkan perusahaan tersebut.

Akibatnya, jika analis membatasi diri dengan melakukan normalisasi data atas satu tabel saja maka terdapat risiko bahwa *database* yang diperoleh hanya memenuhi sebagian kebutuhan sistem informasi akuntansi perusahaan. Seandainya diusahakan penyusunan satu tabel yang mencakup segala kebutuhan perusahaan, maka tabel itu akan menjadi besar sekali (banyak sekali *field*) sehingga untuk memecahnya menjadi beberapa tabel pun menjadi sangat rumit.

Kelemahan normalisasi data, yang tidak menjamin semua kebutuhan data perusahaan tercakup dalam rancangan *database*, dapat diatasi dengan menggunakan ERD. Dengan memanfaatkan ERD, maka seluruh kebutuhan data perusahaan dapat diidentifikasi. Kelemahan dari ERD adalah bahwa bisa terjadi kerancuan antara pendefinisian entitas, *relationship*, dan *attribute* (Date, 1995: 363; Silberschatz, et al., 2002: 37). Sebagai contoh, jika seseorang harus membuat ERD untuk sebuah tempat praktik dokter, apakah resep dan obat merupakan entitas sendiri-sendiri, ataukah obat merupakan atribut dari entitas resep? Demikian pula, apakah sebuah transaksi merupakan *relationship*, ataukah dapat dinyatakan sebagai entitas tersendiri. Misalnya, bila *customer* membeli barang, hal ini dicatat dalam dokumen faktur penjualan. Apakah oleh analis dibuat entitas Customer dan Barang yang dihubungkan oleh *relationship* Membeli? Ataukah dibuat tiga entitas: Customer, Faktur Penjualan, dan Barang, kemudian Customer dan Faktur Penjualan dihubungkan dengan *relationship* Menerima, serta Faktur Penjualan dan Barang dihubungkan dengan *relationship* Mencantumkan?

Tabel 6. Perbandingan Antar Cara Perancangan *Database*

Aspek	Normalisasi Data	ERD	REA
Keunggulan	Mudah, aturan jelas.	Menjamin perancangan *database* yang mencakup seluruh data perusahaan.	Menghilangkan ambiguitas pendefinisian entitas, *relationship*, dan atribut.
Kelemahan	Tidak cocok untuk tabel yang terlalu besar dan kompleks; tidak menjamin seluruh data tercakup dalam rancangan *database*.	Adanya ambiguitas dalam pendefinisian entitas, *relationship*, dan atribut.	Kaku dalam mendefinisikan entitas *event* sehingga memaksa sejumlah dokumen diperlakukan menjadi satu entitas saja.

Dengan pengalaman, seorang analis akan mengetahui alternatif yang mana yang harus dipilihnya dalam penyusunan ERD, namun bagi seseorang yang baru mempelajari ERD, ambiguitas seperti ini akan menyulitkan.

Kesulitan dalam menentukan apakah data merupakan entitas, *relationship,* atau atribut, dapat diatasi dengan menggunakan REA Model. Keunggulan REA Model adalah bahwa apa yang harus dinyatakan sebagai entitas sudah sangat jelas: hanya *resources*, *events*, dan *agents* yang dijadikan entitas. Namun demikian hal ini menimbulkan masalah yang baru lagi, yaitu berkenaan dengan *event*. Dalam hal *resources* dan *agents* hal ini tidak menjadi masalah, tetapi menganggap satu *event* sebagai satu entitas dapat menjadi masalah karena sebuah *event* belum tentu diwakili hanya oleh satu dokumen. Misalnya saja, pada kasus Customer membeli Barang dengan menggunakan dokumen Faktur Penjualan. Bagaimana jika juga dibutuhkan *picking ticket*, *packing slip*, dan surat jalan? Maka sebenarnya kegiatan "membeli" tersebut diwakili tidak saja oleh satu dokumen Faktur Penjualan saja, melainkan oleh empat dokumen terpisah. Memaksakan semua dokumen ini dalam satu entitas akan menyebabkan rancangan *database* menjadi kaku, seolah-olah satu Faktur Penjualan pasti hanya memiliki satu *Picking Ticket*, satu *Packing Slip*, dan satu Surat Jalan. Padahal, sangat dimungkinkan satu Faktur Penjualan membutuhkan beberapa *Picking Ticket* (barang disimpan di beberapa gudang), beberapa *Packing Slip* (barang tidak dapat dibungkus dalam satu kotak), dan beberapa Surat Jalan (barang dikirim dalam beberapa kali pengiriman). Hal ini justru tidak dianjurkan dalam pemodelan REA dan dikategorikan sebagai sebuah *implementation compromise* (Dunn, et al., 2005: 318-320). Tetapi, pada praktiknya memang ada dokumen (oleh Dunn, et al., disebut *task*) yang tidak dapat dimasukkan ke *event* tertentu sehingga tetap terpisah sebagai *task*. Sebagai contoh, sebuah kontra bon adalah *commitment event* yang terkait dengan *cash disbursement event* dan tidak dapat dikaitkan ke dalam *Business Process REA Model* kecuali dengan cara menjadikan *task* sebagai suatu entitas (Goesman, 2011:95). Juga ditemukan bahwa penyusunan rancangan *database* hanya dengan mengandalkan REA Model dapat menghasilkan *database* yang belum dalam kondisi 3NF (Goesman, 2011: 95). Keunggulan dan kelemahan masing-masing cara perancangan *database* ditampilkan dalam Tabel 6.

Perancangan *Database* Menggunakan Kombinasi REA Model, ERD, dan Normalisasi Data

Di dalam makalah ini diajukan sebuah metode perancangan *database* yang menggunakan kombinasi dari tiga cara yang telah dijelaskan, yaitu: REA Model, Entity-Relationship Diagram, dan Normalisasi Data. Metode ini berusaha untuk menghilangkan kelemahan dari satu cara dengan menggunakan kekuatan dari cara yang lain.

Langkah-langkah perancangan *database* yang diusulkan tersebut adalah sebagai berikut:

A. Tahap Pemodelan REA

1. Buat Value System REA Model.
2. Buat Value Chain REA Model.
3. Buat Business Process (BP) REA Model dari masing-masing siklus bisnis menggunakan Diagram Kotak.
    a. Buat Core BP REA Model.
    b. Kembangkan dengan Instigation Event, Commitment Event, dan Reversal Event menjadi Extended BP REA Model.
    c. Hilangkan Resources atau Agents yang tidak akan dicatat.

B. Tahap Pemodelan Entity-Relationship (ERD)

4. Untuk setiap Event pada BP REA Model, tentukan dokumen-dokumen yang diperlukan.
5. Berdasarkan BP REA Model, buat ERD yang belum dinormalisasi.
    a. Setiap Resource menjadi Entity.
    b. Setiap Event diwakili oleh dokumen-dokumennya. Setiap dokumen itu menjadi Entity tersendiri.
    c. Setiap Agent menjadi Entity.
    d. Tentukan attributes masing-masing entity.
    e. Tentukan relationships di antara entities yang ada.
    f. Tentukan cardinality masing-masing relationship.

C. Tahap Normalisasi Data

6. Buat ERD yang telah dinormalisasi. Lakukan normalisasi data atas entities yang belum dalam kondisi 3NF. Usahakan semua *relationship* sudah 1-M, tidak ada lagi M-M atau 1-1. Relasi M-M ditemukan jika salah satu entitas masih mengandung *repeating group.* Relasi tersebut akan otomatis terganti oleh dua relasi 1-M setelah dilakukan pemisahan tabel untuk menghilangkan *repeating group.* Seandainya masih ditemukan relasi M-M meskipun entitasnya tidak mengandung *repeating group*, maka buatlah tabel penghubung antara kedua entitas yang bersangkutan. Sedangkan untuk menghilangkan relasi 1-1 adalah cukup sederhana, yaitu dengan cara menggabungkan kedua entitas yang terkait menjadi satu entitas saja.

D. Tahap Penyelesaian

7. Lakukan **integrasi** dengan ERD dari *business process cycles* yang lain.
    a. Integrasikan melalui entitas Resource, Event, atau Agent yang sama.
    b. Perbaiki ERD yang dihasilkan dengan mengupayakan penyelesaian atas:
        i. Name conflict
        ii. Attribute conflict
        iii. Cardinality conflict

8. Buat *logical database* berdasarkan ERD yang telah diperoleh.

Logika yang mendasari pengembangan metode ini berlandaskan pada kekuatan masing-masing cara. REA Model adalah satu-satunya dari ketiga cara yang telah dibahas yang memberi gambaran perusahaan dengan lingkungannya (dalam bentuk *value system REA Model*) serta siklus-siklus beserta interaksinya di dalam perusahaan (*value chain REA Model*). Kedua diagram itu bermuara pada *Business Process REA Model* di mana diyakinkan bahwa tidak ada *Resources, Events*, atau *Agents* yang relevan bagi perusahaan yang dilupakan untuk dicatat. Secara praktis, REA Model ini memudahkan perancang *database* untuk dapat mengidentifikasi entitas-entitas untuk keperluan pengembangan ERD.

Salah satu kesulitan yang paling dirasakan oleh mereka yang baru mulai menyusun ERD, adalah menentukan entitas-entitasnya. Dengan bantuan BP REA Model, maka entitas-entitas itu teridentifikasi dengan lebih mudah, yaitu semua *Resources*, semua *Agents*, dan semua *Events.* Khusus untuk *events* diwakili oleh dokumen-dokumen yang dipakai untuk mencatat *events* yang bersangkutan.

ERD yang dikembangkan pertama kali oleh perancang *database* biasanya tidak berada dalam kondisi yang sudah memenuhi syarat-syarat normalisasi data. Oleh karena itu, atas ERD yang pertama kali dihasilkan harus dilaksanakan proses normalisasi terlebih dahulu sehingga semua entitas dalam ERD tersebut sudah berada dalam kondisi 3NF.

Sebagai langkah terakhir baru dilakukan integrasi antar ERD dari berbagai siklus yang berhubungan (misalnya, antara *revenue cycle* dan *expenditure cycle*) serta disusun rancangan *database* secara logis.

Penerapan Metode Kombinasi Pada Perusahaan

Penerapan metode kombinasi dilaksanakan pada sebuah studi kasus yang, meskipun fiktif, telah diusahakan serealistis mungkin.

P.T. Ayam Bebek Puyuh (PT. ABP) adalah sebuah perusahaan yang berlokasi di Bandung. Bidang usahanya adalah sebagai pedagang besar telur ayam, telur bebek, dan telur puyuh, yang disuplai ke perusahaan-perusahaan seperti halnya hotel, restoran, dan toko-toko makanan. Perusahaan ini tidak berskala terlampau besar, namun memiliki posisi likuiditas yang sangat baik karena RUPS telah memutuskan untuk tidak pernah menggunakan dana pinjaman (kredit) untuk membiayai operasi perusahaan.

Telur didatangkan menggunakan truk milik PT. ABP dari peternakan-peternakan di Jawa Tengah. Ketika tiba di lokasi PT. ABP, telur-telur itu dikelompokkan berdasarkan kualitasnya, kemudian ditempatkan dalam *tray*, di mana satu *tray* dapat menampung 30 butir telur ayam atau telur bebek.

Sedangkan untuk telur puyuh, satu *tray* dapat menampung 100 butir telur puyuh. Karena produk yang ditanganinya sangat mudah pecah, maka seluruh pekerjaan pengelompokan dan penempatan dalam *tray* dilakukan secara manual, tanpa mesin. Untuk itu, PT. ABP mempekerjakan 15 orang karyawan.

Pelanggan PT. ABP memesan telur melalui telepon pada saat membutuhkannya dan dalam jumlah yang dibutuhkannya. *Customer order* tersebut diterima oleh PT. ABP dan kemudian dibuatkan *Sales Order* sebanyak tiga lembar. Lembar ketiga berfungsi sebagai *picking ticket* ke gudang. Setelah barang disiapkan, maka lembar ini disimpan sebagai arsip di kantor PT. ABP. Sedangkan lembar ke-1 dan ke-2 dibawa oleh pengirim bersama telur yang dipesan ke alamat pelanggan. Semua pesanan dikirim menggunakan mobil milik PT. ABP (perusahaan ini tidak melayani pembeli yang datang di lokasi PT. ABP). Ketika sampai, pelanggan memeriksa kondisi telur, kemudian SO ke-1 dan ke-2 itu ditandatangani oleh pelanggan. SO ke-1 dibawa kembali ke kantor PT. ABP, sedangkan lembar ke-2 disimpan pelanggan.

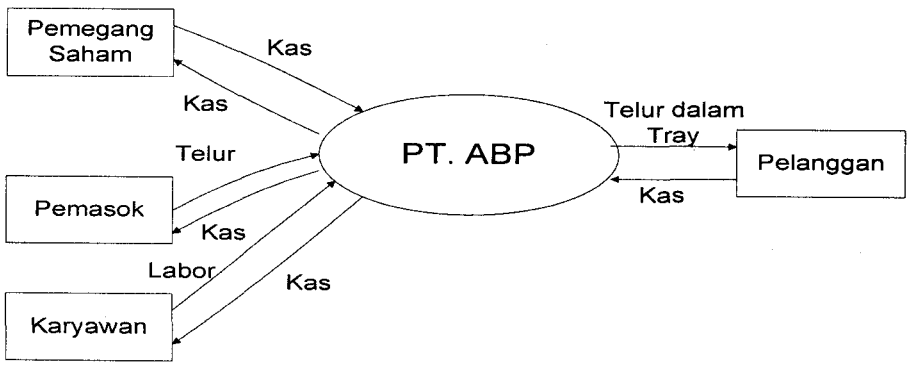
Penagihan ke pelanggan dilakukan sebulan sekali. Prosedurnya adalah sebagai berikut: pelanggan menjumlahkan nilai semua SO ke-2 yang ada padanya, sementara PT. ABP menjumlahkan nilai semua SO ke-1 yang ada di arsipnya. Kemudian kedua nilai tersebut dicocokkan. Jika sudah cocok, maka PT. ABP membuat kuitansi sesuai dengan nilai jumlah tersebut. Pelanggan menyerahkan uang pembayaran beserta semua SO ke-2, dan menerima kuitansi dan semua SO ke-1 sebagai gantinya. Oleh PT. ABP, SO ke-2 diarsip bersama dengan SO ke-3 yang telah disimpannya.

Berikut adalah penerapan metode kombinasi REA Model – ERD – Normalisasi Data atas kasus PT. ABP.

A. Tahap Pemodelan REA

1. Buat *Value System REA Model*

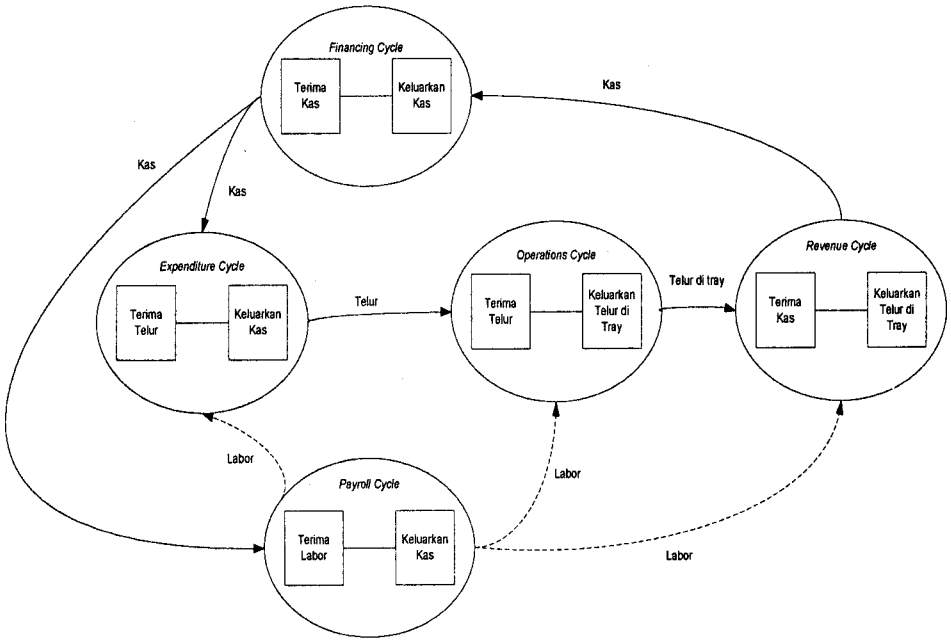
Pada tahap ini, disusun sebuah *Value System REA Model*. Model tersebut bagi PT. ABP dapat dilihat dalam Gambar 4



Gambar 4. PT. ABP *Value System REA Model*

2. Buat *Value Chain REA Model*

Value Chain REA Model untuk PT. ABP telah disusun dan dapat dilihat dalam Gambar 5.

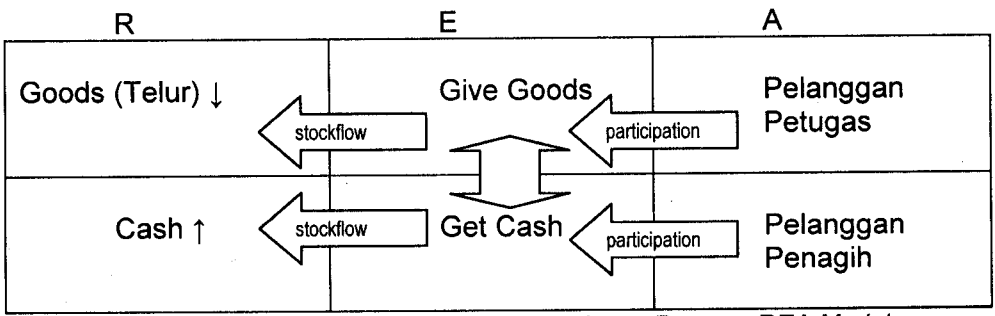


Gambar 5. PT. ABP *Value Chain REA Model*

Catatan: Panah yang menghubungkan *Payroll Cycle* digambarkan terputus-putus, karena meskipun Karyawan menyumbangkan tenaga (*labor*) pada siklus-siklus yang lain, hal tersebut tidak dicatat secara eksplisit - dalam arti, tidak dilakukan pencatatan identitas karyawan yang melakukan pekerjaan tertentu baik pada saat perolehan telur, penempatan di *tray*, maupun penyerahan telur di *tray* kepada pelanggan.

3. Buat *Business Process REA Model* menggunakan Diagram Kotak.

Dalam contoh ini yang dibuat adalah BP REA Model untuk *Revenue Cycle* dari PT. ABP. Hal ini dapat dilihat dalam Gambar 6.



Gambar 6. PT. ABP *Revenue Cycle Business Process REA Model*

Karena sistem PT. ABP cukup sederhana, maka tidak perlu dikembangkan menjadi *extended BP REA Model.*

Resources yang tidak akan dicatat adalah *Cash*, karena pada perusahaan ini diasumsikan bahwa tidak dilakukan pencatatan terus menerus atas saldo uang yang dimiliki perusahaan. Maka dari itu *Resources cash* dapat dihilangkan.

Agents yang tidak akan dicatat adalah karyawan PT. ABP, karena itu untuk *Agents* hanya tinggal Pelanggan yang harus dicatat.

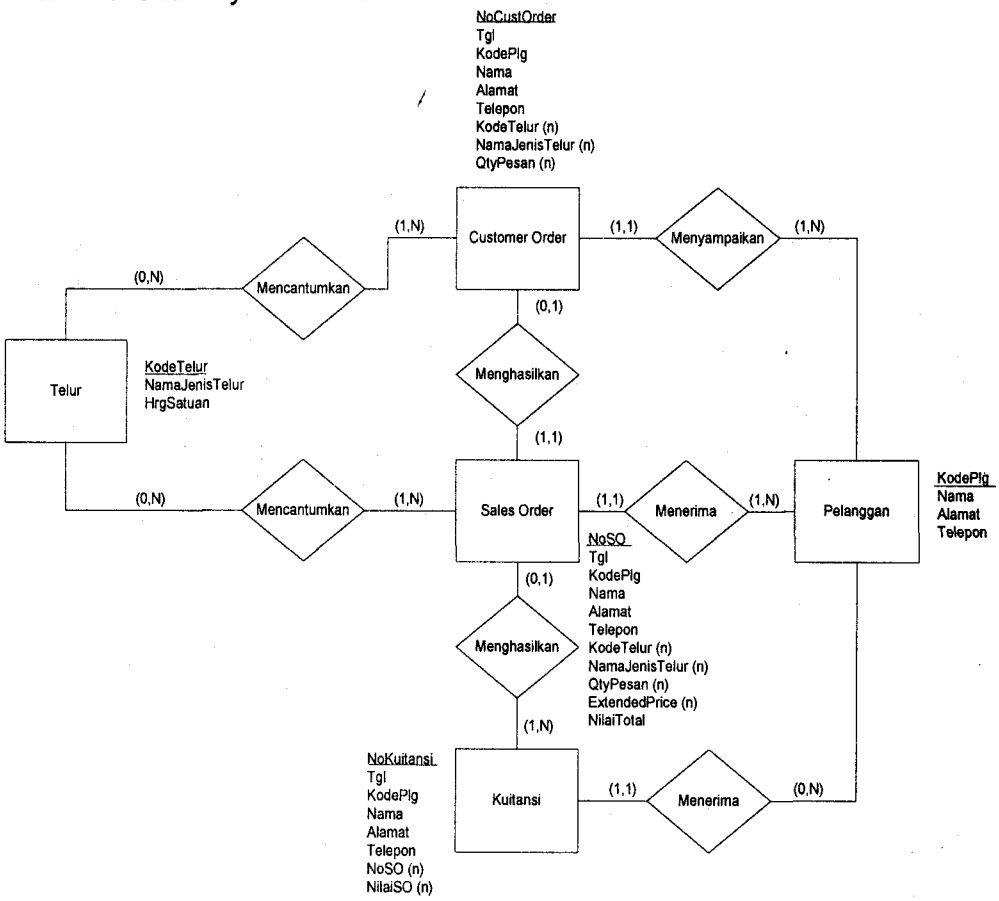
B. Tahap Pemodelan Menggunakan ERD

4. Untuk setiap *Event* pada *BP REA Model*, harus ditentukan dokumen-dokumennya yang terkait.

Pada *Event Give Goods,* maka dokumen yang terkait adalah *Customer Order* dan *Sales Order*. Sedangkan pada *Event Get Cash*, dokumen yang terkait adalah Kuitansi.

5. Berdasarkan *BP REA Model*, buat ERD yang belum dinormalisasi.

Dalam Gambar 7 ditampilkan ERD yang belum dinormalisasi untuk *Revenue Cycle* PT. ABP.

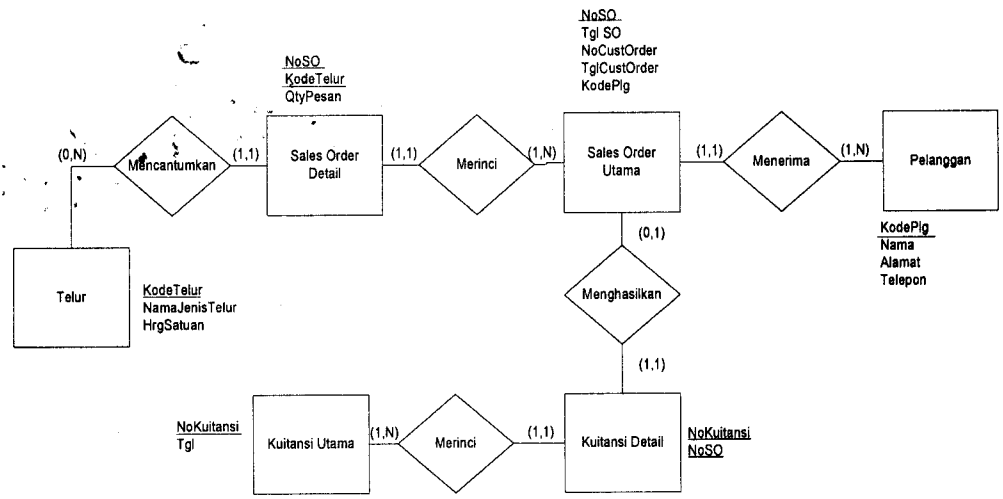


Gambar 7. ERD Yang Belum Dinormalisasi

C. Tahap Normalisasi Data

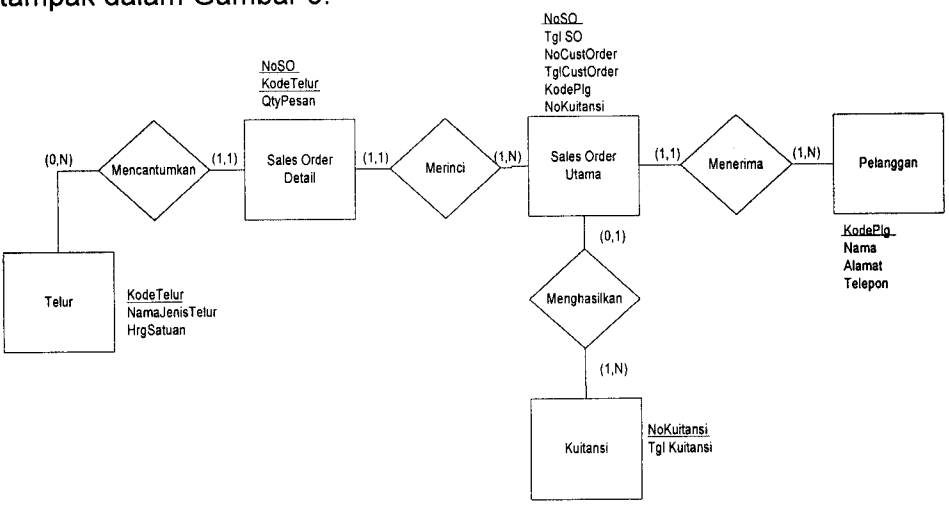
6. Buat ERD yang sudah dinormalisasi. Juga usahakan agar semua relasi menjadi 1-M, tidak ada lagi M-M atau 1-1.

Usaha pertama untuk menghilangkan 1-1 (antara *Customer Order* dan *Sales Order*), serta M-M (antara Telur dan *Sales Order*), menghasilkan Gambar 8. Gambar itu juga menunjukkan upaya menjadikan setiap entitas berada dalam kondisi 3NF.



Gambar 8. Normalisasi Data Atas ERD (a)

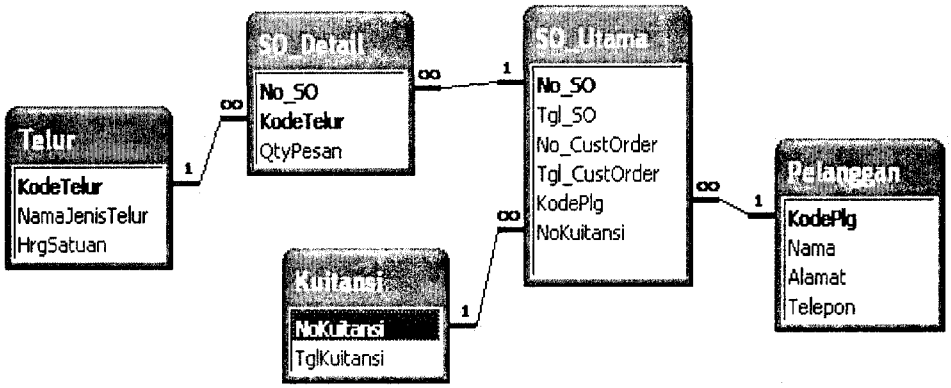
Ternyata usaha menghilangkan *repeating group* pada Kuitansi, menyebabkan munculnya lagi relasi 1-1 antara SalesOrderUtama dengan KuitansiDetail. Oleh karena itu, kedua entitas tersebut disatukan, sehingga hasil akhir dari ERD yang telah dinormalisasi adalah sebagai tampak dalam Gambar 9.



Gambar 9. Normalisasi Data Atas ERD (b)

D. Tahap Penyelesaian

Buat *logical database* berdasarkan ERD yang diperoleh. Konversi dari ERD menjadi *logical database* bagi PT. ABP adalah sebagai tampak dalam Gambar 10. Setelah dikaji, ternyata semua tabel sudah dalam kondisi 3NF sehingga rancangan *logical database* itu sudah selesai.



Gambar 10. *Logical Database* Untuk *Revenue Cycle* PT. ABP

Kesimpulan

Setelah diujicoba menggunakan sebuah studi kasus, maka akhirnya dapat disimpulkan bahwa metode kombinasi REA Model, ERD, dan normalisasi data memang dapat dipergunakan untuk membuat rancangan *logical database* yang benar. Sejumlah temuan yang didapat selama uji coba tersebut dapat dipergunakan untuk menyempurnakan metode ini, yaitu:

a. *Value System REA Model* dan *Value Chain REA Model,* meskipun dapat membantu dalam pengembangan *Business Process REA Model,* ternyata tidak mutlak diperlukan karena keterikatan antara dua model pertama dengan yang terakhir agak renggang alias tidak terkait secara erat. Oleh karena itu pengguna dapat langsung mengadaptasi REA Model (Diagram Kotak) ke dalam bentuk *Business Process REA Model* tanpa menghadapi risiko bahwa *database* yang dirancangnya menjadi tidak benar.
b. Analisis yang efektif dapat dilakukan langsung atas kalimat-kalimat dalam contoh kasus, tanpa perlu menyusun *Business Process REA Model* dalam bentuk Diagram Kotak. Caranya adalah dengan memecah setiap kalimat menjadi komponen-komponen R - E - A – nya. Hal ini membantu dalam pemahaman, meskipun kebingungan masih mungkin terjadi dalam membedakan "data" dengan "proses". Kebingungan ini terutama dirasa saat ada kalimat mengenai proses yang tidak melibatkan data.
c. Berbeda dengan ERD klasik, di mana baik *entities* maupun *relationships* dapat memiliki atribut, serta kedua-duanya adalah kandidat untuk dijadikan tabel dalam *logical database*, maka ternyata

dengan menggunakan metode ini dapat dipastikan bahwa semua tabel pasti berasal dari *entities*. Demikian pula, semua atribut pasti terdapat pada *entities* dan tidak pernah terdapat pada *relationships*. Jadi *relationships* tidak akan pernah menjadi tabel dan tidak akan pernah mengandung atribut. Dengan demikian, adalah kurang relevan untuk memberi nama pada *relationships* sehingga *relationships* tidak perlu diberi nama.

d. Pengguna harus memahami saat menyatukan dua atau lebih entitas yang memiliki relasi 1-1, yaitu entitas mana yang "dilebur" ke dalam entitas yang lainnya. Misalnya pada kasus yang telah dibahas, perlu dipahami mengapa *Customer Order* yang dilebur ke dalam *Sales Order* dan bukan sebaliknya.

e. Kasus yang telah dibahas juga menunjukkan keunggulan ERD atas normalisasi data yang belum disebut dalam Tabel 6. Hal ini berkenaan dengan entitas Kuitansi yang tadinya terhubung dengan entitas Pelanggan (lihat Gambar 7) namun pada *logical database* hubungan itu sudah tidak ada lagi, melainkan terjadi melalui tabel SO_Utama (lihat Gambar 10). Jika Kuitansi dinormalisasi tanpa melihat ke ERD, maka masih akan terdapat *field* KodePlg di tabel Kuitansi. Namun dengan adanya ERD, maka tampak bahwa KodePlg tidak dibutuhkan lagi di tabel Kuitansi, dan justru penambahan field itu akan menimbulkan risiko inkonsistensi antara KodePlg di tabel SO_Utama dengan KodePlg di tabel Kuitansi.

Referensi:

Date, C. J. (1995), *An Introduction To Database Systems*, 6th edition, Massachusetts: Addison-Wesley Publishing Company.

Dunn, C. L.; Cherrington, J. O.; and Hollander, A. S. (2005), *Enterprise Information Systems: A Pattern-based Approach*, 3rd edition, New York: McGraw-Hill International Edition.

Goesman, S. (2011), *Peranan Rancangan Database Dalam Pengambilan Keputusan Mengenai Persediaan*, Skripsi yang tidak dipublikasikan, Bandung: FE Unpar.

Hall, J. A. (2008), *Accounting Information Systems*, 6th edition, Mason: Cengage Learning.

Kendall, K. E., and Kendall, J. E. (2005), *Systems Analysis and Design*, 6th edition, New Jersey: Pearson/Prentice-Hall International Edition.

Romney, M. B., and Steinbart, P. J. (2009), *Accounting Information Systems*, 11th edition, New Jersey: Pearson International Edition.

Silberschatz, A; Korth, H. F.; and Sudarshan, S. (2002), *Database System Concepts*, 4th edition, Boston: McGraw-Hill.

Whitten, J. L. And Bentley, L. D. (2008), *Introduction to Systems Analysis & Design*, 1st Edition, New York: McGraw-Hill Irwin.