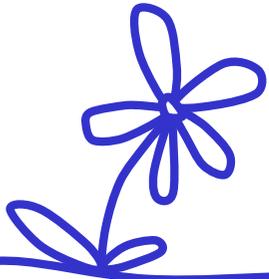


ざっくり始める
データモデリング

2010/11/12 おーがみ

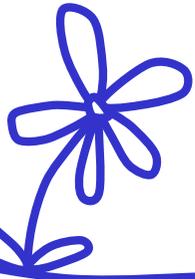




自己紹介

- 👉 ビズアイユ こと おーがみ。
- 👉 自称、さすらいのデータアーキテクト。
- 👉 結構シブなDB2エキスパート資格持ち。
(↑でも、業務では全然触ってない…)
- 👉 残りの情報は二次会でどうぞ。

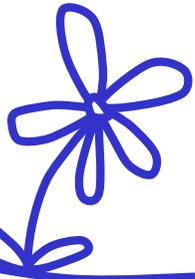




モデリングとは？

- 👉 対象を**構造化**し、**関連**を作図すること。
- 👉 モデルにより、**業務**や**処理**を俯瞰できる。
- 👉 モデルは、**概念**/**論理**/**物理**の粒度を持つ。
- 👉 モデリングには**様々な図法**がある。
(↑ ERD、ORD、THD、UML、DFD…)





開発におけるモデルの必要性

👁 システムの全体像が見えるようになる。

☆ 詳細な部分も明確になる。

👁 ユーザーとの合意形成に利用し易い。

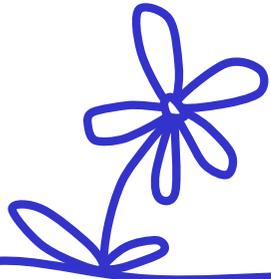
👁 生産性が向上する機能を利用できる。

☆ コードやDDLを自動生成する。

☆ 既存プログラムやDBをリバースする。

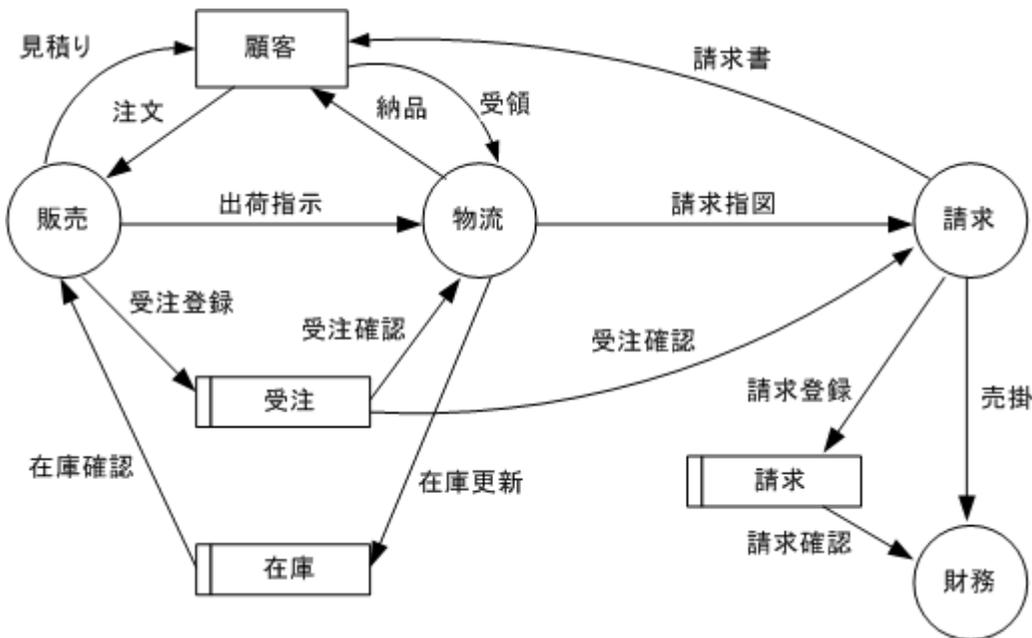
☆ モデルの修正を全体に波及させる。





代表的なモデリング図法

👁️ Data Flow Diagram

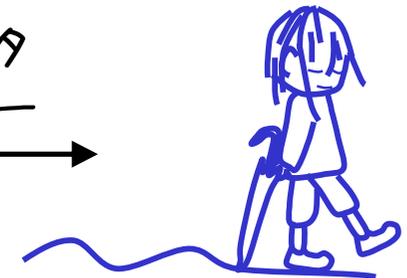


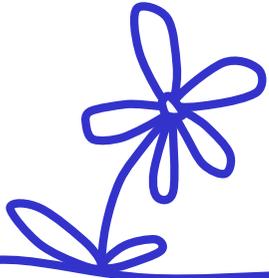
データの発生・吸収

処理

データストア

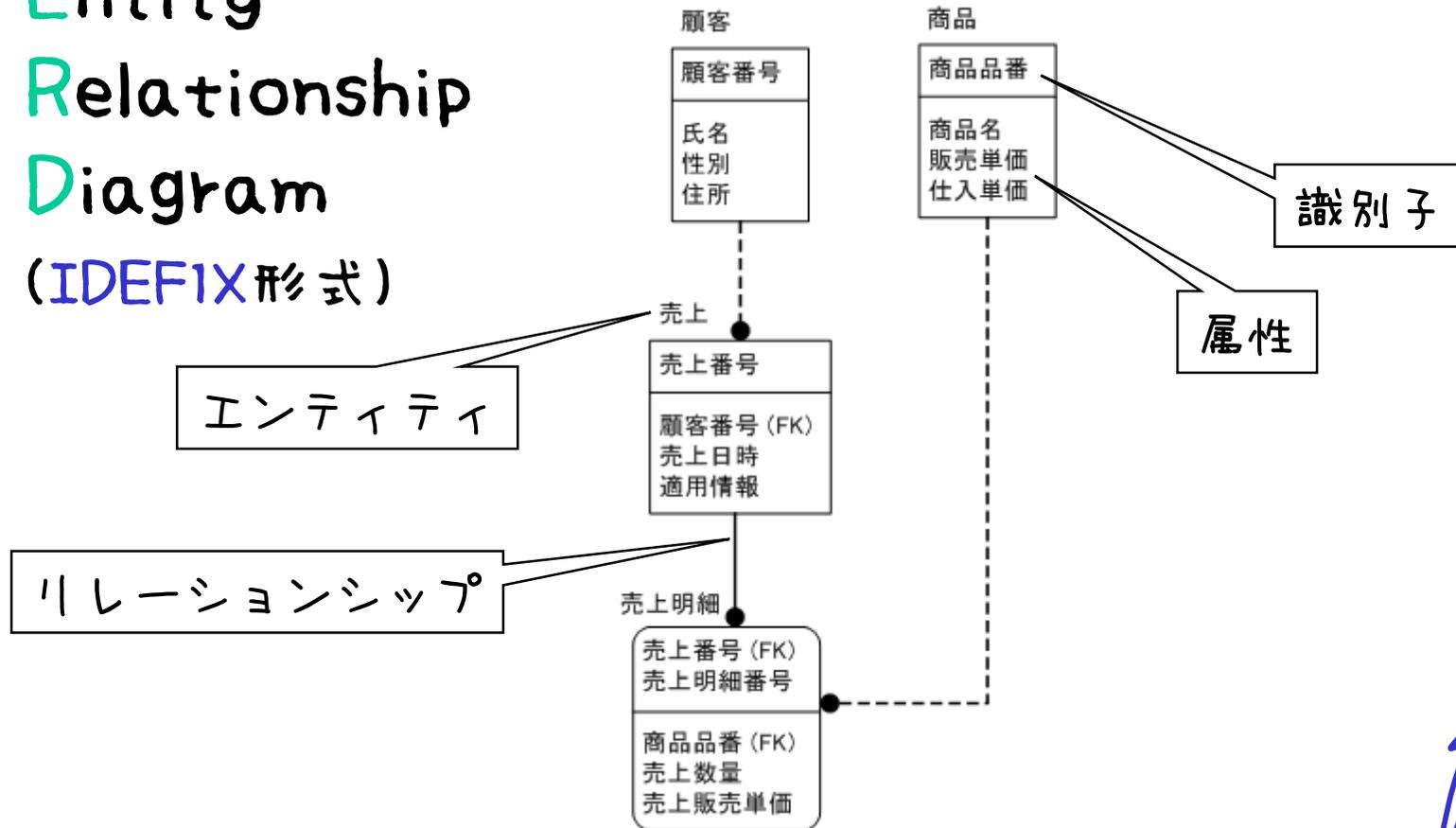
データフロー

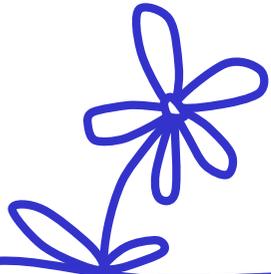




代表的なモデリング図法

Entity Relationship Diagram (IDEFIX形式)

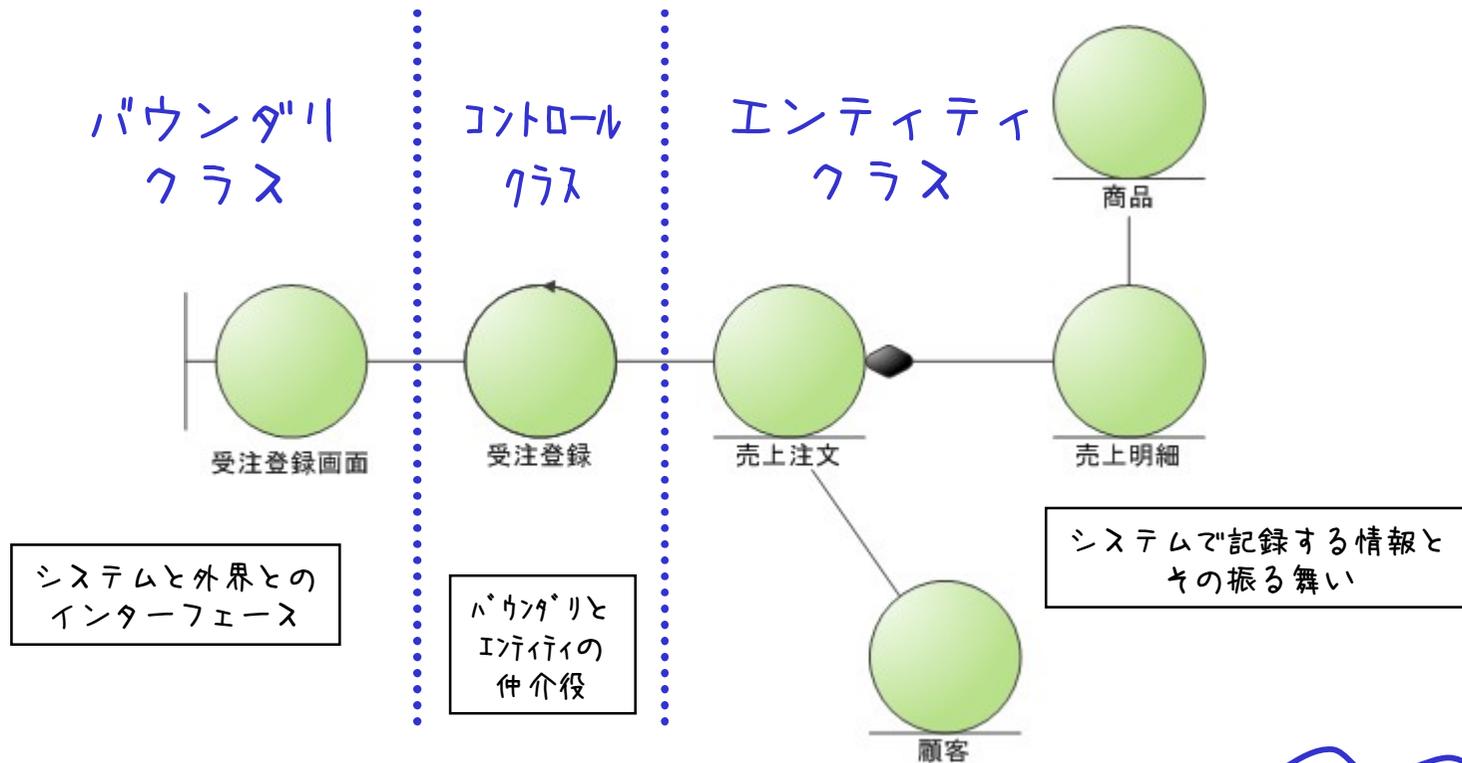


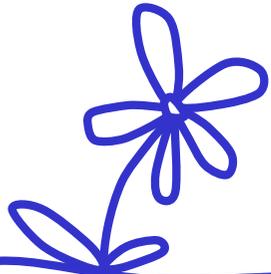


代表的なモデリング図法

Robustness Diagram

(ロバストネス分析 + UMLのクラス図)





図法の違い = 考え方の違い

⑤ Process Oriented Approach

DFD

☆業務に着目して設計を行う考え方

⑤ Data Oriented Approach

ERD

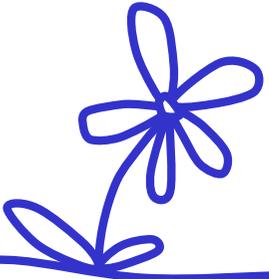
☆データ構造に着目して設計を行う考え方

⑤ Object Oriented Approach

UML

☆データとその挙動を一つのオブジェクトと見なすことで設計を行う考え方





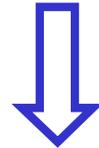
POAの弱点

👉 業務に着目して設計を行うと…

☆データの重複や不整合が起こり易い。

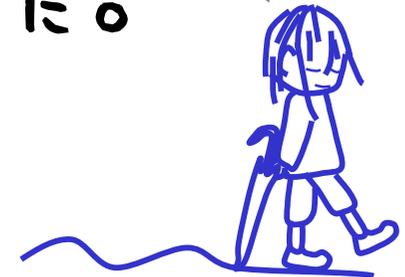
☆業務が変化すると、プログラムとデータの両方が修正対象になる。

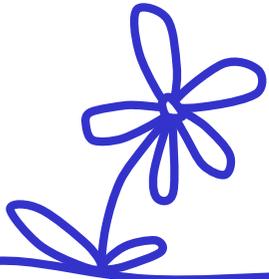
☆システムが縦割りになり易い。



Dead On Arrival?

この反省としてDOAが提唱された。





DOAの良いところ

⑤ 整理されたデータモデルを作成できる。

☆ 変化を受け入れられる構造が望ましい。

⑤ データモデル上のシステムであれば…

☆ データの重複や不整合を防げる（正規化）。

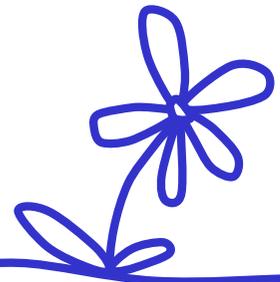
☆ 業務の変化からデータ構造が守られる。

☆ コードのI/O化により保守が容易になる。

☆ 横断的にデータを分析し易くなる。

データこそ最大の情報資源！

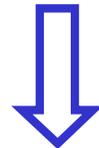




DOAの弱点

👁 データ構造に着目して設計を行うと…

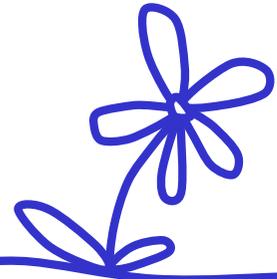
☆ データ構造を変化させた際に影響を受ける
プログラムを特定するのが非常に難しい。



OOAが普及する理由の一つとなった。

とはいえ、変化に脆弱なデータ構造に
プログラムを紐付けてしまうと、
プログラムも脆弱になってしまう。
つまり、DOAが機能してOOAも機能する！





ERDでデータモデリング

👉 何故ERDでデータモデリングなのか？

☆表記法が分かり易い。

☆データ構造の表現力に富む。

☆関係データベースとの親和性が高い。

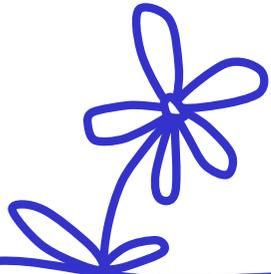
☆多くの人々が利用している。

☆書籍やWebに教材がたくさんある。

(↑玉石混交な点には要注意。)

階層、ネットワーク、関係、オブジェクト、多次元、XML等の
データベースモデルとデータモデルは別物です！





ざっくりとした

ERDを書くためのポイント

1. エンティティ（テーブル）を洗い出す。

☆ イベントとリソースの違いに注意すべし！

2. 属性（列）を埋めていく。

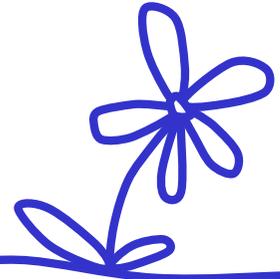
☆ 主キーには代理キーを使おう！

3. 重複を排除するために正規化する。

☆ 一つの場所に一つの事実…事実とは何か？

(One fact in One Place)





エシキなERDを書いてみる

毎度ご利用いただきありがとうございます

また、お知らせ
金融機関やコンビニが料金を収納する
ことにはありません。

電気ご使用量のお知らせ 大神 貴謙 様

ご使用場所 エチオピア アムハラ州 バハルダール市 ミレニアム・カフェ

22年10月分	ご使用期間 9月 7日～10月 6日 検針月日 10月 7日 (30日間)	ご契約種別 従量電灯B
ご使用量	159kWh	ご契約 30A
請求予定金額 (うち消費税等相当額)	3,640円 173円	当月指示数 6033 前月指示数 5874 差引 159
基本料金 819円00銭		計器乗率(倍) 取替前計量値 計器番号(下3桁) 781
電力量料金 ・1段料金 2,144円40銭 ・2段料金 891円54銭 ・燃料費調整額 -214円65銭 太陽光促進付加金 0円		ムダなくムリなく 暮らしの省エネルギー 太陽光促進付加金単価(1kWhあたり) 0銭 燃料費調整のお知らせ(1kWhあたり)
		10月(当月)分 -1円35銭 11月(翌月)分 -1円60銭 翌月分は当月分比べ -0円25銭

次回検針予定日 11月 8日

地区番号	お客さま番号
04	49272-30504-1-04

検針員 工藤

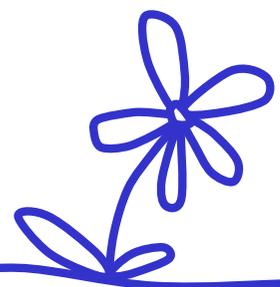
お問い合わせ先/カスタマーセンター
お引越し、ご契約の変更
0120-995-005
その他の電気に関するご用件
0120-995-006

東京電力株式会社
銀座支社(001)
TEPCO

～CO₂に関する情報等は裏面をご覧ください～

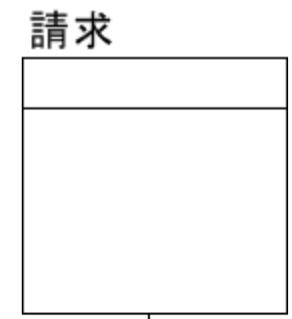
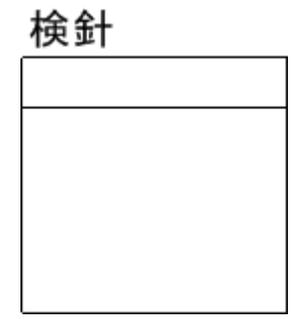
A5:SQLというツールが
作図に超便利でした!
(IE形式)





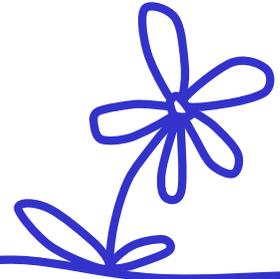
1. イベントを見出す

👉 帳票の目的を表現する「~する」が
イベントのエンティティとなる。



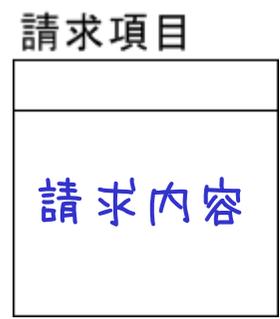
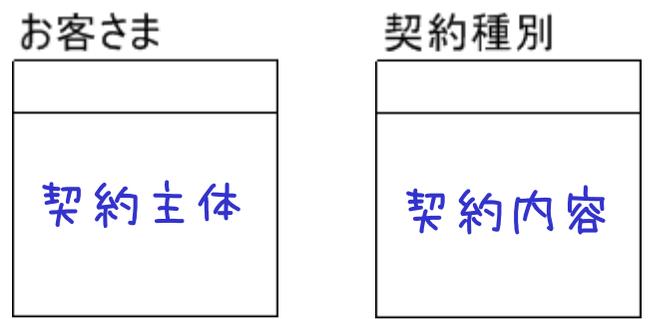
「契約した内容」と、
「検診した結果」と、
「請求する金額」の
お知らせでしたね。





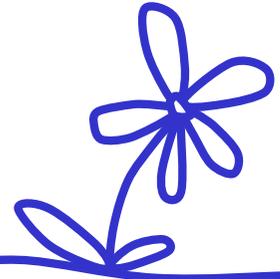
2. リソースを抜き出す

イベントに紐付く「誰が/何を/どこに」
がリソースのエンティティとなる。



いくつかの細かいリソースについては割愛しましたよー。





3. 属性を埋めていく

お客さま

お客さまID
お客さま番号
名前
地区番号

お客さま番号をそのまま主キーにすると、
コード体系が変わったときに大変です。

契約

契約ID
ご使用場所
計器番号

検針

検針ID
検針日
指示数

請求

請求ID
支払期限日

契約種別

契約種別ID
契約種別名
契約アンペア
基本料金

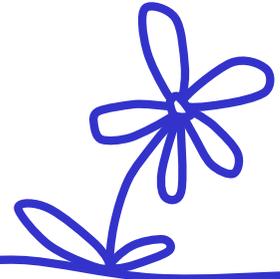
請求内訳

請求内訳ID
請求ID (FK)
料金

請求項目

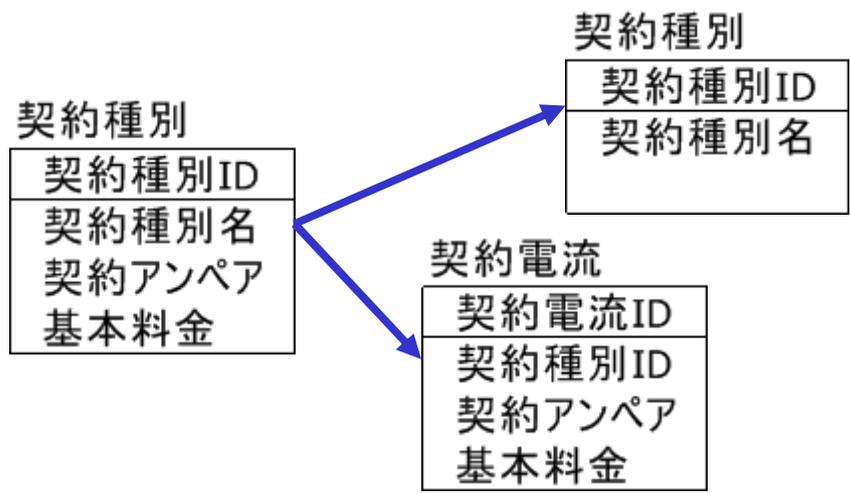
請求項目ID
請求項目名





4. 正規化する

属性を確認し、重複が起こる部分は別のエンティティに切り出す。



スタンダードなメニューなら

- 従量電灯B
 - 計算する
 - 契約概要
- 従量電灯C
 - 計算する
 - 契約概要

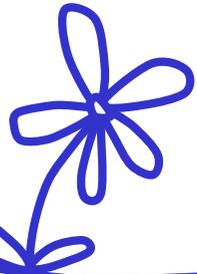
夜間がトクいなお客さまなら

- おトクなナイト8
 - 計算する
 - 契約概要
- おトクなナイト10
 - 計算する
 - 契約概要



同じ契約アンペアでも、契約種別ごとに基本料金が違います。





なぜ正規化を行うのか？

- ≡ 構文論的 (Syntactic) な視点：
素直な関係演算を可能にするため。
(↑ NULL を取り除くことも含む。)

Not NF

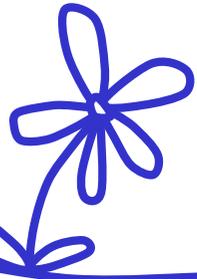
Account	Amount #1	Amount #2	Amount #3	Amount #4
A	200	300	1,000	400
B	600			
C	1,000	200		

5NF

Account	Line	Amount
A	1	200
A	2	300
A	3	1,000
A	4	400
B	1	600
C	1	1,000
C	2	200

500円以上の商品が
購入された取引明細を
求めるSQLを考えると...





なぜ正規化を行うのか？

意味論的 (Semantic) な視点：

一つの関係に対応するエンティティを一つに限定し、更新時異状を防ぐため。

1NF

Emp ID	Emp Name	Post ID	Post Name
001	Korematsu	001	President
101	Tanaka	100	Manager
102	Sato	110	Senior
103	Inoue	111	Associate
104	Suzuki	111	Associate

従業員エンティティと
役職エンティティが
関係内に混在したのが
更新時異状の原因

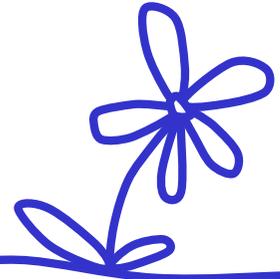
5NF

Emp ID	Emp Name	Post ID
001	Korematsu	001
101	Tanaka	100
102	Sato	110
103	Inoue	111
104	Suzuki	111

5NF

Post ID	Post Name
001	President
100	Manager
110	Senior
111	Associate





第一正規形のおさらい

定義：関係Rの全ての属性が
 スカラ値（単一）であり、
 繰り返し項目を持たない。

Not NF

ID	First Name	Family Name	Telephone Number
1	Robert	Ingram	555-861-2025
2	Jane	Wright	555-403-1659, 555-403-1660, 555-776-4100
3	Maria	Fernandez	555-808-9633

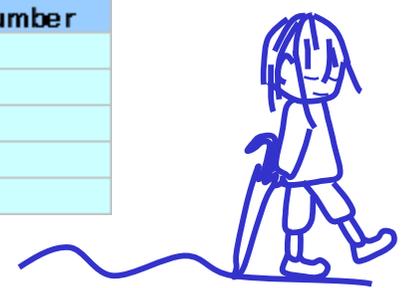
Date氏的には
 これも非正規形

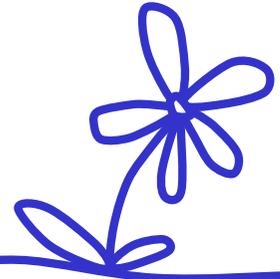
ID	First Name	Family Name	Tel #1	Tel #2	Tel #3
1	Robert	Ingram	555-861-2025		
2	Jane	Wright	555-403-1659	555-403-1660	555-776-4100
3	Maria	Fernandez	555-808-9633		

Not NF

1NF

ID	Line	First Name	Family Name	Telephone Number
1	1	Robert	Ingram	555-861-2025
2	1	Jane	Wright	555-403-1659
2	2	Jane	Wright	555-403-1660
2	3	Jane	Wright	555-776-4100
3	1	Maria	Fernandez	555-808-9633





第二正規形のおさらい

定義：第一正規形の条件を満たし、
 全ての非キー属性が候補キーに
 部分関数従属していない。

1NF

Employee	Skill	Department	Current Work Location
Jones	Typing	General Affairs	114 Main Street
Jones	Shorthand	General Affairs	114 Main Street
Jones	Whittling	General Affairs	114 Main Street
Bravo	Light Cleaning	Production	73 Industrial Way
Ellis	Alchemy	Production	73 Industrial Way
Ellis	Flying	Production	73 Industrial Way
Harrison	Light Cleaning	Production	73 Industrial Way

候補キーが複数の属性で
 構成される場合のみ
 部分関数従属が起こる

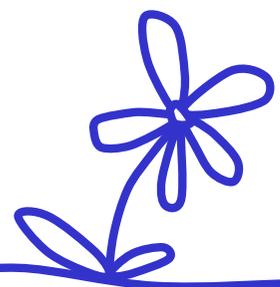
2NF

Employee	Department	Current Work Location
Jones	General Affairs	114 Main Street
Bravo	Production	73 Industrial Way
Ellis	Production	73 Industrial Way

Employee	Skill
Jones	Typing
Jones	Shorthand
Jones	Whittling
Bravo	Light Cleaning
Ellis	Alchemy
Ellis	Flying
Harrison	Light Cleaning

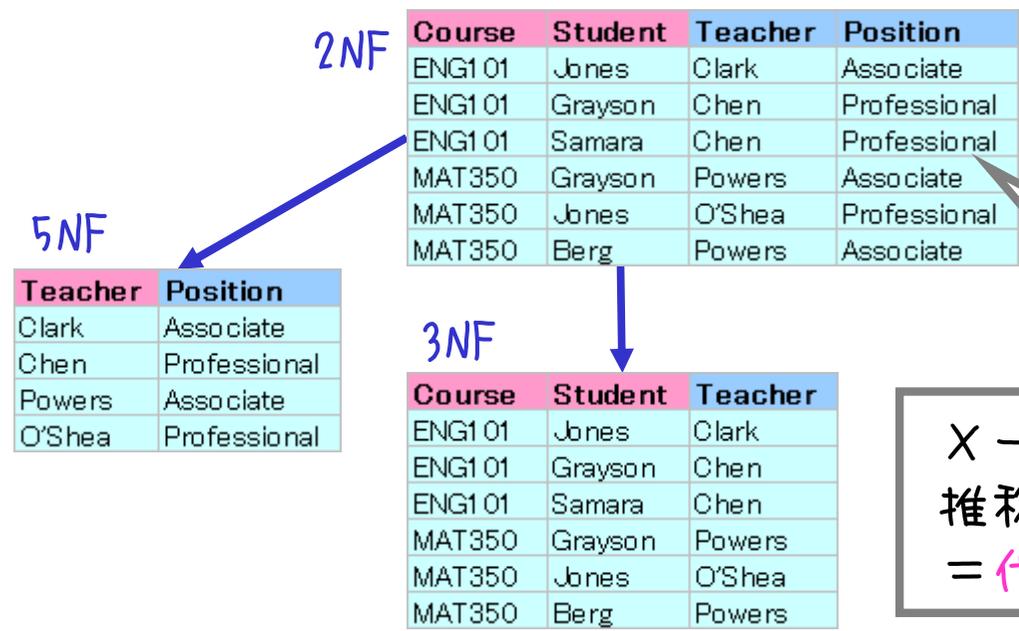
5NF





第三正規形のおさらい

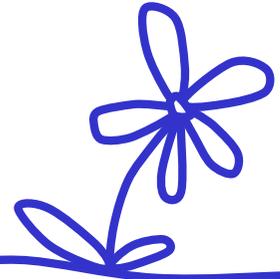
定義：第二正規形の条件を満たし、
 全ての非キー属性が候補キーに
 推移的関数従属していない。



非キー属性間に関数従属
 = 推移的関数従属アリ

$X \rightarrow Y \rightarrow Z$ でも $Y \rightarrow X$ なら
 推移的関数従属にならない
 = 代理キーは使ってもOK





Boyce-Codd正規形のおさらい

定義：第三正規形の条件を満たし、
 非キー属性から候補キーの
 一部分に対する関数従属がない。

{講座、学生} → 講師
講師 → 講座

Course	Student	Teacher
ENG101	Jones	Clark
ENG101	Grayson	Chen
ENG101	Samara	Chen
MAT350	Grayson	Powers
MAT350	Jones	O'Shea
MAT350	Berg	Powers

結合従属性を維持して
正規化を行うには
関係を三つに分ける

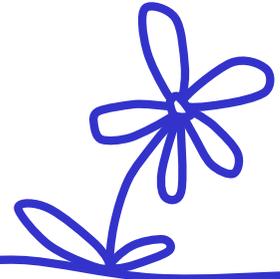
Course	Student
ENG101	Jones
ENG101	Grayson
ENG101	Samara
MAT350	Grayson
MAT350	Jones
MAT350	Berg

Student	Teacher
Jones	Clark
Grayson	Chen
Samara	Chen
Grayson	Powers
Jones	O'Shea
Berg	Powers

Teacher	Course
Clark	ENG101
Chen	ENG101
Powers	MAT350
O'Shea	MAT350

元の関係にあった
関数従属性
(ビジネスルール)
は失われる

BCNF (5NF)



属性の従属性のおさらい

⑤ 関数従属性：

属性Xの値が決まると
属性Yの値も一意に定まる性質。

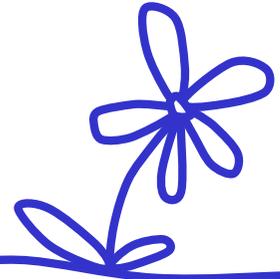
⑤ 多値従属性：

属性Xの値が決まると
属性Yの値も複数に定まる性質。

⑤ 結合従属性：

属性Xによる結合を行っても
属性Yの情報量が増加しない性質。





第四正規形のおさらい

定義：BCNFの条件を満たし、
 対象性のある（複数の）
 多値従属性を含まない。

BCNF

Restaurant	Pizza Variety	Delivery Area
A1 Pizza	Thick Crust	Springfield
A1 Pizza	Thick Crust	Shelbyville
A1 Pizza	Thick Crust	Capital City
A1 Pizza	Stuffed Crust	Springfield
A1 Pizza	Stuffed Crust	Shelbyville
A1 Pizza	Stuffed Crust	Capital City
Elite Pizza	Thin Crust	Capital City
Elite Pizza	Stuffed Crust	Capital City
Vincenzo's Pizza	Thick Crust	Springfield
Vincenzo's Pizza	Thick Crust	Shelbyville
Vincenzo's Pizza	Thin Crust	Springfield
Vincenzo's Pizza	Thin Crust	Shelbyville

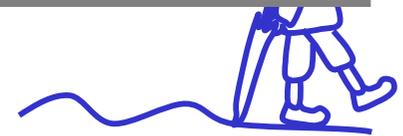
Restaurant	Delivery Area
A1 Pizza	Springfield
A1 Pizza	Shelbyville
A1 Pizza	Capital City
Elite Pizza	Capital City
Vincenzo's Pizza	Springfield
Vincenzo's Pizza	Shelbyville

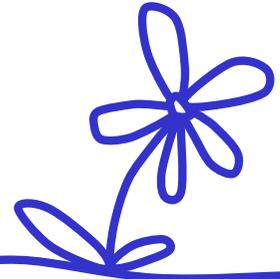
レストランによって
 配達できる地域が
 複数に定まる

Restaurant	Pizza Variety
A1 Pizza	Thick Crust
A1 Pizza	Stuffed Crust
Elite Pizza	Thin Crust
Elite Pizza	Stuffed Crust
Vincenzo's Pizza	Thick Crust
Vincenzo's Pizza	Thin Crust

レストランによって
 作るピザの種類が
 複数に定まる

4NF





第五正規形のおさらい

定義：BCNFの条件を満たし、
 自明でない結合従属性を含まない。

BCNF

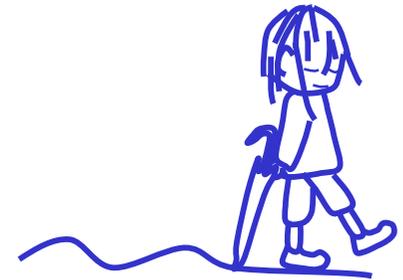
Restaurant	Pizza Variety	Delivery Area
A1 Pizza	Thick Crust	Springfield
A1 Pizza	Thick Crust	Shelbyville
A1 Pizza	Thick Crust	Capital City
A1 Pizza	Stuffed Crust	Springfield
A1 Pizza	Stuffed Crust	Shelbyville
A1 Pizza	Stuffed Crust	Capital City
Elite Pizza	Thin Crust	Capital City
Elite Pizza	Stuffed Crust	Capital City
Vincenzo's Pizza	Thick Crust	Springfield
Vincenzo's Pizza	Thick Crust	Shelbyville
Vincenzo's Pizza	Thin Crust	Springfield
Vincenzo's Pizza	Thin Crust	Shelbyville

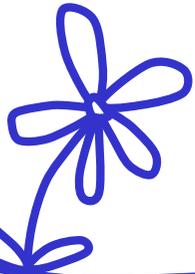
Restaurant	Delivery Area
A1 Pizza	Springfield
A1 Pizza	Shelbyville
A1 Pizza	Capital City
Elite Pizza	Capital City
Vincenzo's Pizza	Springfield
Vincenzo's Pizza	Shelbyville

Pizza Variety	Delivery Area
Thick Crust	Springfield
Thick Crust	Shelbyville
Thick Crust	Capital City
Stuffed Crust	Springfield
Stuffed Crust	Shelbyville
Stuffed Crust	Capital City
Thin Crust	Capital City
Thin Crust	Springfield
Thin Crust	Shelbyville

Restaurant	Pizza Variety
A1 Pizza	Thick Crust
A1 Pizza	Stuffed Crust
Elite Pizza	Thin Crust
Elite Pizza	Stuffed Crust
Vincenzo's Pizza	Thick Crust
Vincenzo's Pizza	Thin Crust

5NF





数学的な正規化のまとめ

👉 候補キーを単一の属性で構成すると、関係をシンプルに正規化できる。

☆ 候補キーが単一の属性の場合、

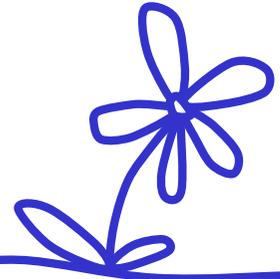
Not NFか 2NFか 5NFにしかない。

👉 エンティティの連関を一つの関係に複数盛り込むと、話が厄介になる。

☆ 二つの属性で構成された候補キーが

組(行)である関係は必ず 5NFとなる。

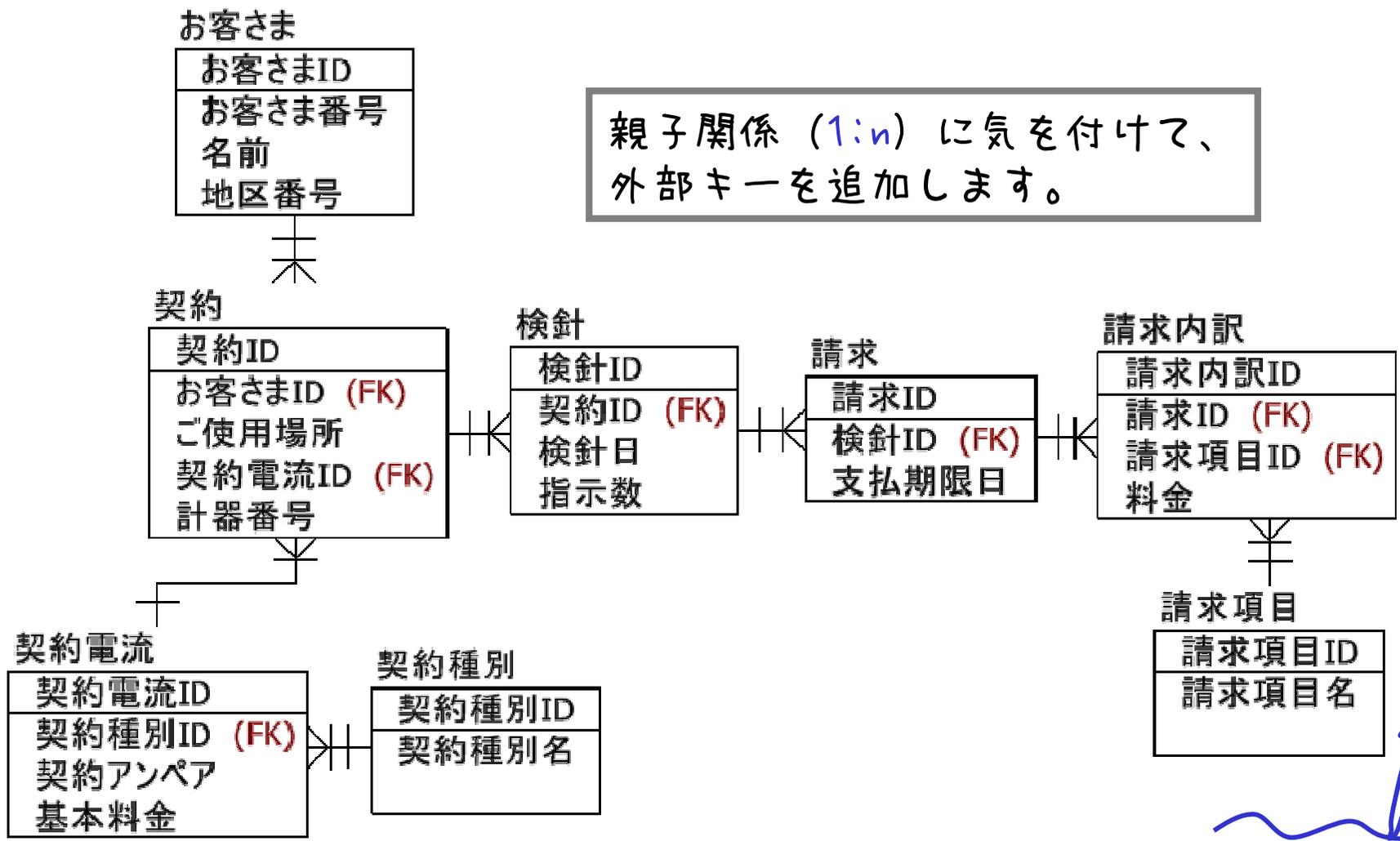


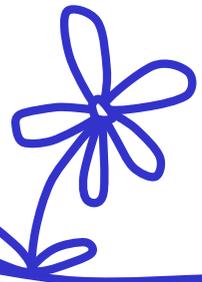


リレーションシップ

5. 関連を設定する

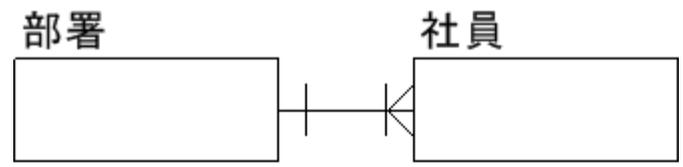
親子関係 (1:n) に気を付けて、外部キーを追加します。



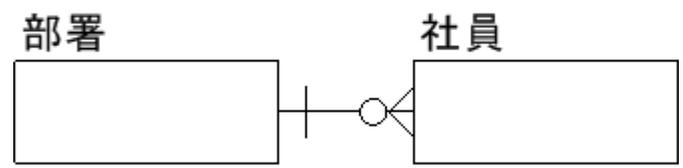


ビジネスルールの取り扱い

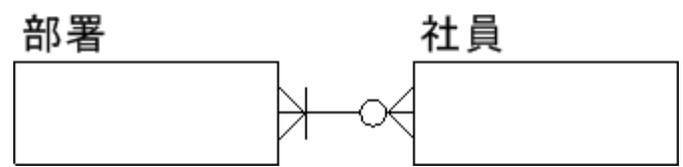
リレーションシップの表記によって
様々なビジネスルールを表現できる。



社員は一つの部署に所属する。
部署には所属する社員が必ずいる。



社員は一つの部署に所属する。
所属する社員がいない部署もある。



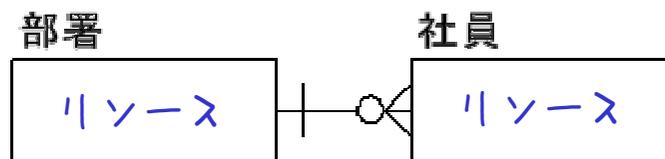
社員は一つ以上の部署に所属する。
所属する社員がいない部署もある。



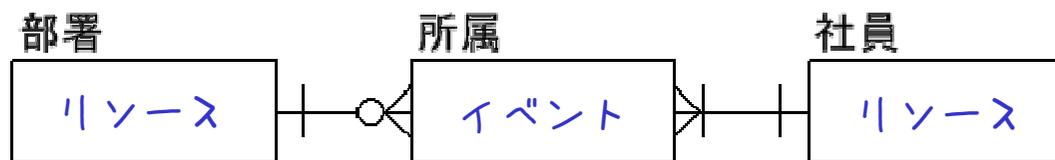
🌸 連関エンティティを意識する

👉 多対多の関連をそのまま関係モデルで表すと、非正規形になってしまう。

☆これを防ぐため、**連関エンティティ**を作る。

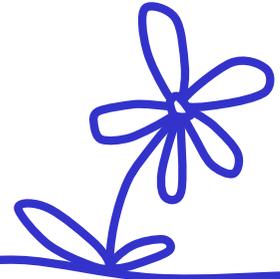


リソースとリソースが関連する場合、たとえ1対多でも連関エンティティが必要な場合が多いので注意すべし！



「所属する」という業務の明確な姿を表している。
(所属の履歴も追跡し易くなっている。)





リレーションシップ

5. 関連を設定する (修正版)

オプションナリティ (○) を考慮した、完成品です。

お客さま

お客さまID
お客さま番号
名前
地区番号



契約

契約ID
お客さまID (FK)
ご使用場所
契約電流ID (FK)
計器番号

検針

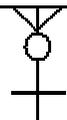
検針ID
契約ID (FK)
検針日
指示数

請求

請求ID
検針ID (FK)
支払期限日

請求内訳

請求内訳ID
請求ID (FK)
請求項目ID (FK)
料金



請求項目

請求項目ID
請求項目名

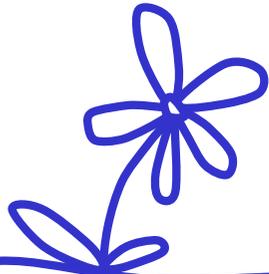
契約電流

契約電流ID
契約種別ID (FK)
契約アンペア
基本料金

契約種別

契約種別ID
契約種別名





中華なERDを書いてみよう

TAKE OUT お持ち帰りご注文用紙 **電話予約 OK!!**

このマークのお料理は、お持ち帰りもできます。この用紙にご記入の上、お気軽にご注文ください。

商品名	金額	ご注文数	商品名	金額	ご注文数
お持ち帰り専用プレート			海老のチリソース		
わくわくプレートA(3~4人前)	¥2,500(原¥2,625)		前菜	¥699(原¥733)	
わくわくプレートB(3~4人前)	¥2,900(原¥3,045)		甘海老フライの金沙炒(赤唐辛子)	¥199(原¥208)	
わくわくプレートC(3~4人前)	¥3,400(原¥3,570)		チャーシュー・メンマ盛り	¥299(原¥313)	
点心			若鶏のからあげ(5コ)	¥299(原¥313)	
山盛りポテトフライ	¥299(原¥313)		W若鶏のからあげ(10コ)	¥579(原¥607)	
焼餃子	¥190(原¥199)		【新名物】上海チキン	¥380(原¥399)	
W焼餃子	¥350(原¥367)		飯・麺		
海老春巻	¥169(原¥177)		五目焼そば	¥699(原¥733)	
W海老春巻(2本)	¥299(原¥313)		海老とこのカレービーフン	¥699(原¥733)	
チャーシューちまき	¥180(原¥189)		チャーハン	¥449(原¥471)	
一品料理			レタスチャーハン	¥549(原¥576)	
特製おぼろ豆腐のkok旨マーボー	¥399(原¥418)		天津飯	¥499(原¥523)	
激辛おぼろの四川マーボー	¥399(原¥418)		中華丼	¥699(原¥733)	
蒸し鶏とピーマンの旨塩炒め	¥469(原¥492)		豚ニラ焼肉ご飯	¥699(原¥733)	
豚肉と豆腐の野菜旨煮	¥476(原¥499)		ズワイ蟹のあんかけチャーハン	¥679(原¥712)	
海老マヨネーズ	¥476(原¥499)		特製牛あんかけごはん	¥699(原¥733)	
香港風酢豚	¥476(原¥499)		小ごはん	¥119(原¥124)	
ニラたっぷり野菜炒め	¥499(原¥523)		ごはん	¥199(原¥208)	
ホイコーロウ	¥499(原¥523)		大盛りごはん	¥249(原¥261)	
若鶏と甘酢しょうゆ	¥499(原¥523)		デザート		
豚肉とホクラゲの玉子炒め	¥499(原¥523)		ごま付きだんご	¥249(原¥261)	
ころころ根菜の黒酢豚	¥539(原¥565)		はちみつ揚げパン	¥249(原¥261)	
牛肉とピーマンの細切り炒め	¥599(原¥628)				
海鮮と野菜の黒胡椒炒め	¥599(原¥628)		おつまみ辣油	¥199(原¥208)	

■ご記入の上、係の者にお渡しください。
 ●お電話でもご注文できます。指定の時間までにお作りしておきます。
 ※注文数が多い場合は前日までにご連絡ください。
 ※混雑時は、時間がかかる場合もございますのでご了承ください。一部メニューは変更になる場合もございます。
 時間帯によって販売していない商品がございます。

お名前 _____ お電話番号 _____

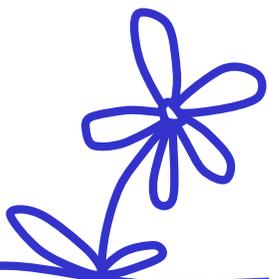
CHINESE RESTAURANT
パ・ニヤン
 水天宮前店
 東京都中央区日本橋人形町1-11-203号
 TEL.03-5649-8068

パ・ニヤンのテイクアウトメニュー
TAKE OUT 出来たてを **電話予約 OK!!**
お持ち帰り!

クーポン券で商品の値引がある。
 深夜だと料金の加算がある。
 …という設定で考えてください。

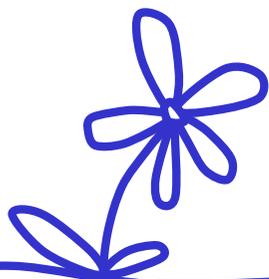
ズワイ蟹のあんかけチャーハン
 ¥679(原¥712)



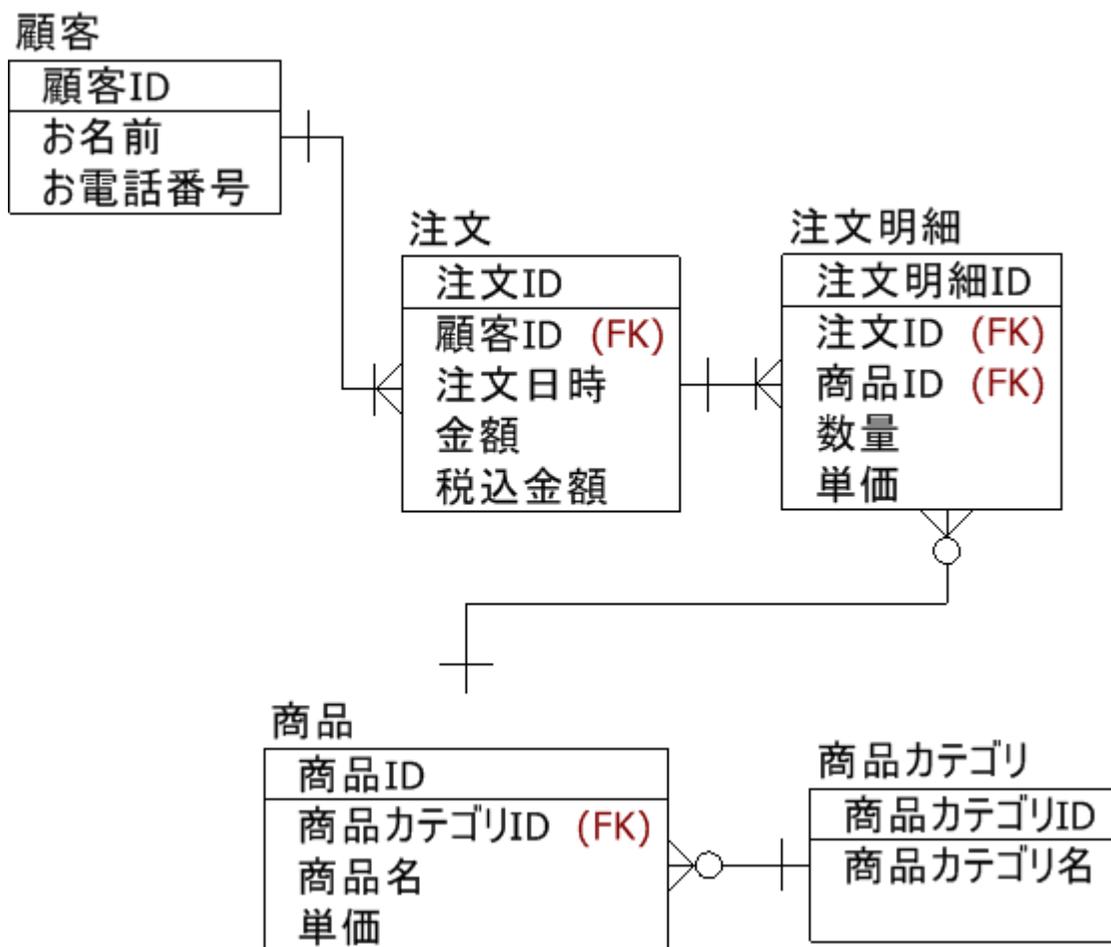


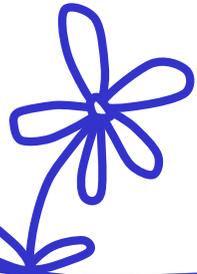
このページに作図してね





僕の書いたERDはこれだ

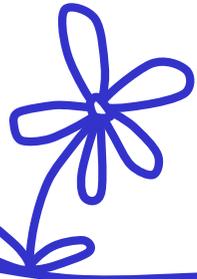




前ページERDの特徴

- 👉 クーポン券による値引に対応するため、
注文明細にも単価を持たせた。
- 👉 深夜料金の加算に対応するため、
注文金額は導出項目にしなかった。
- 👉 消費税率が変わっても、その時点での
税込金額は変化しないようにした。
(↑ 税込金額を属性として持たせた。)



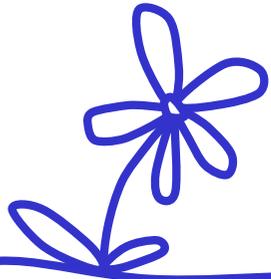


業務的な正規化の考え

- 👉 同じ名前の属性でも、意味に違いがあるなら一つにまとめてはいけない。
- 👉 排除しても良い導出項目は、逆導出が可能なものだけである。
- ☆ 逆導出が可能な導出項目は、論理モデルでビューに盛り込む。

複雑な業務を持つ場合、
データモデルも複雑なのです。
諦めてその通りに作図しましょう。





僕の修正したERDはこれだ

顧客

顧客ID
お名前
お電話番号

注文

注文ID
顧客ID (FK)
注文日時
注文総額
税込総額

注文明細

注文明細ID
注文ID (FK)
商品ID (FK)
注文数量
注文単価

商品

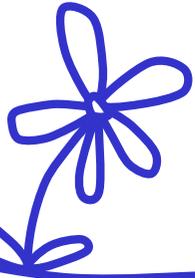
商品ID
商品カテゴリID (FK)
商品名
商品単価

商品カテゴリ

商品カテゴリID
商品カテゴリ名

同音異義語 (ホモニム) と
異音同義語 (シノニム) は
使わないようにすべし!

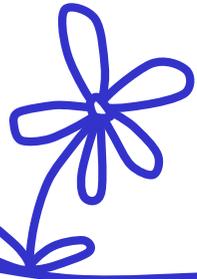




まとめ

- 👉 データモデルで、データの構造だけではなく業務の姿まで表現できる。
(↑特にT字形ER手法で顕著。)
- 👉 エンティティと属性を洗い出せば、技法に従って良質なERDを作れる。
- 👉 ビジネスルールを確認できるスキルが、モデルの精度を決めるカギとなる。





でも、まだ終われない

👉 DAの仕事は果てしなく続く...

☆ CRUD (Create, Read, Update, Delete) 分析

☆ トランザクションの正規化

☆ DLC (Data Life Cycle) 分析

☆ 論理データモデルの作成

» リレーショナルモデル

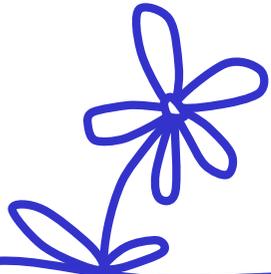
» OLAPモデル

» BIGDATAモデル

☆ 物理設計、運用設計、パフォーマンス

今までのERモデルは
概念データモデル。





論理データモデル色々

👉 リレーショナルモデル

☆ 関係データベースへの実装を前提とし、
ERモデルに編集を加えたもの。

(↑非正規化*やビューの追加。) ※非推奨

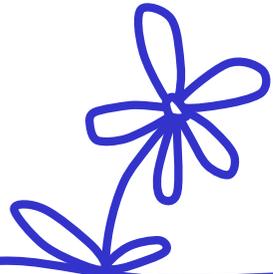
👉 OLAPモデル

☆ 多次元分析の実施を前提として作られた、
スタースキーマなどの分析用モデル。

👉 BIGDATAモデル

☆ クラウド環境で前提となるデータモデル。





OLAPモデルのスキーマ例

スタースキーマ

時間ディメンジョン

時間ID
日
週
月
四半期
半期
年度

売上ファクト

セルID
時間ID (FK)
商品ID (FK)
顧客ID (FK)
部署ID (FK)
売上数量
売上金額

商品ディメンジョン

商品ID
商品コード
商品名
商品小分類
商品中分類
商品大分類

顧客ディメンジョン

顧客ID
顧客コード
顧客名
顧客所在県
顧客所在地方

部署ディメンジョン

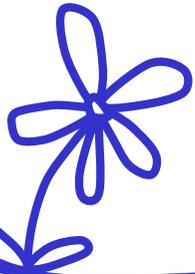
部署ID
部署コード
部署名
部署区分

大福帳スキーマ

売上明細

明細ID
日
週
月
四半期
半期
年度
商品名
商品小分類
商品中分類
商品大分類
顧客名
顧客所在県
顧客所在地方
部署名
部署区分





ちゃぶ台を返してみる

👁️ 今までの説明には **デタラメ** も…

☆ エンティティとテーブルを **混同** させている。

(↑ ERモデルと Relationalモデルは別物。)

☆ 代理キーの利用は **実装** 上の話である。

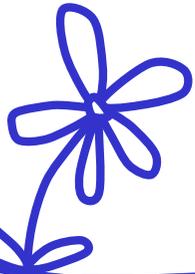
(↑ 実はモデリングとは関係ない。)

☆ エンティティの **サブセット** に触れていない。

(例：従業員のパートと正社員の区分)

「ざっくり」を卒業して、
「しっかり」考えなくちゃ！





参考文献

👁️ 入門者向け

☆プロとしてのデータモデリング入門

» 長所：ざっくりデータモデリングを抑えられる。

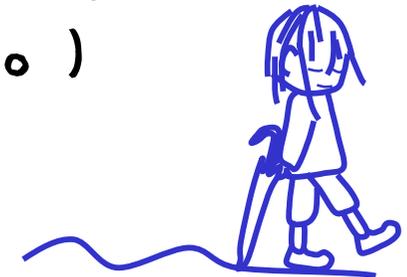
» 短所：用語や方法論に不正確な部分がある。

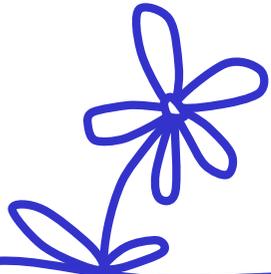
☆情報処理教科書：DBスペシャリスト

» 長所：一通りRDB周りの理論を抑えられる。

» 短所：問題を解かないと説明が頭に入らない。

(説明が簡潔にまとめられ過ぎ。)





参考文献

👁️ 初心者向け

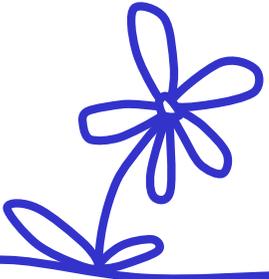
☆楽々ERDレッスン

- » 長所：現場レベルのデータモデリングが分かる。
- » 短所：実践第一なので、正確な理論は後回し。

☆ビジネス環境の変化に強いDB設計

- » 長所：最上流から物理設計まで連続した方法論。
- » 短所：T字形ER手法についての言及が無い。
(正規化簡便法の元ネタなのは明確。)





参考文献

👁️ 中級者向け

☆UMLによる一気通貫DBシステム設計

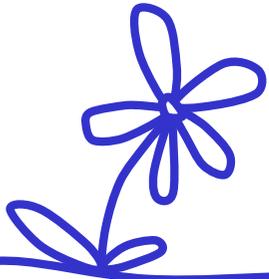
- » 長所：OOAでのデータモデリングが分かる。
- » 短所：一つ一つのトピックが若干消化不良。

☆データベース設計論・T字形ER

- » 長所：厳密な概念データモデルを実現できる。
- » 短所：内容が難解。部分的適用では火傷する。

T字形ER手法については、
今後掘り下げてみる予定です。





グループはこちら

👉 ブログ:

流離と見聞録

👉 mixi と gree と twitter:

全部 **ビズアイユ** で検索

👉 接近戦:

結構な割合で **ClubDB2** にいる

