情報セキュリティマネジメント

試験対策テキスト

Information
Security
Management
Exam

無料体験入学者用

TAC

本書に記載されている会社名または製品名は，一般に各社の商標または登録商標です。
なお，本書では，各社の商標または登録商標については ® および ™ を明記していません。

はじめに

ITの専門家だけが情報セキュリティを考える時代は終わってしまいました。私たちの何気ない行動やミスが，セキュリティ事故に発展して莫大な被害が発生するようなことも，十分に起こり得るのです。今や情報セキュリティは，ITを使用する私たち全員が意識しなければならない課題なのです。

そのようなニーズにこたえるべく，2016年(平成28年)に情報セキュリティマネジメント試験が始まりました。ITの専門家に限らず「仕事でITに触れる人」に向けた資格試験で，社会人の必須スキルとして，あるいはこれから社会人を目指す人の武器として，大いに活用できるはずです。

本書はその情報セキュリティマネジメント試験に合格するためのテキストです。覚えるべき知識は整理すると共に，表面的な理解に終わらないように説明すべき点はしっかり説明することを心掛けています。重要なポイントは，演習問題や事例で振り返ることができるようにも配慮しました。本書を活用して，ぜひ合格を勝ち取っていただきたいと思います。

TAC情報処理講座

目　次

0. 情報セキュリティとは 1

第1部　情報セキュリティ分野 5

1. 情報セキュリティ技術 7

1.1 情報セキュリティの基本概念 8
1.2 サイバー攻撃と情報セキュリティ対策 13
1.3 情報セキュリティを確保する技術 44
1.4 情報セキュリティ教育，普及啓発活動 67
1.5 確認演習 70

2. 情報セキュリティ管理 73

2.1 情報セキュリティ管理とISMS 74
2.2 ISMSの確立 78
2.3 ISMSの実行 88
2.4 ISMSの監視とレビュー 90
2.5 ISMSの維持と継続的改善 92
2.6 セキュリティインシデントの管理 93
2.7 確認演習 97

3. 情報セキュリティ対策の実践 99

3.1 標的型攻撃による情報の漏えい 100
3.2 不正アクセスによるWebサイトの改ざん 104
3.3 エクスプロイト 107
3.4 クラウド利用におけるデータ消失・流出 110
3.5 スマートデバイスからの情報の漏えい 113
3.6 内部不正による情報の漏えい 116
3.7 インターネットバンキングなどによる金銭被害 120
3.8 確認演習 123

4. 関連法規 127
4.1 セキュリティ関連法規 128
4.2 知的財産権関連法規 136
4.3 労働関連・取引関連法規 140
4.4 その他 145
4.5 確認演習 147

5. 確認演習問題の解答解説 149

第2部 関連分野 157
6. 関連分野 157
6.1 システム構成要素 158
6.2 データベース 167
6.3 ネットワーク 177
6.4 プロジェクトマネジメント 191
6.5 サービスマネジメント 201
6.6 システム監査 212
6.7 システム戦略 219
6.8 システム企画 225
6.9 企業活動 230

索引 243

第2部
関連分野

情報セキュリティマネジメント試験の午前試験では，情報セキュリティ分野の知識に加え，関連分野の知識に関する問題も出題されます。

関連分野は，データベースやネットワークなどの応用技術や，プロジェクトマネジメント，企業活動などのマネジメント系・ストラテジ系の分野など，多岐にわたります。

第2部では，これらの関連分野の基礎知識を紹介します。午前試験対策として活用してください。また，情報セキュリティ分野の学習で登場した用語が分からないときなどのリファレンスとしても利用してください。

なお本書では，以降の内容は第1部の第5章から続く第6章として，“6.1”などのように記したうえで各知識を解説します。

6. 1 システム構成要素

【Guide】
情報システムにはどんな処理形態があるのか，性能や信頼性を高めるにはどんな技術を用いるのかについて学びましょう。

(1)　システムの処理形態・利用形態

①　処理形態の分類

業務システムの処理形態は，処理のタイミングによって次のように大別できます。

■バッチ処理

処理すべきデータを蓄積しておき，特定のタイミングで一括して処理します。給与計算や売上高の集計，バックアップの取得などに適しています。

■リアルタイム処理

処理要求が発生するたびに，そのつど処理を行います。主に次の二つがあります。

・オンライントランザクション処理(OLTP：On-Line Transaction Processing)

トランザクション(一連の処理単位)がサーバなどに届くたびに処理するような形態です。銀行のATM端末における処理や座席予約などに適しています。

・リアルタイム制御処理

微小な時間単位で状況を監視しながら，事象の発生(状況の変化)に応じて迅速に対応を行う形態です。工場用ロボットの自動運転の制御や家電製品に組み込まれたマイコンなどに用いられています。

リアルタイム処理のうち，「入力 → 結果表示 → 次の入力 → …」のように，要求と応答を繰り返しながら処理を進めるものを，対話型処理ということがあります。

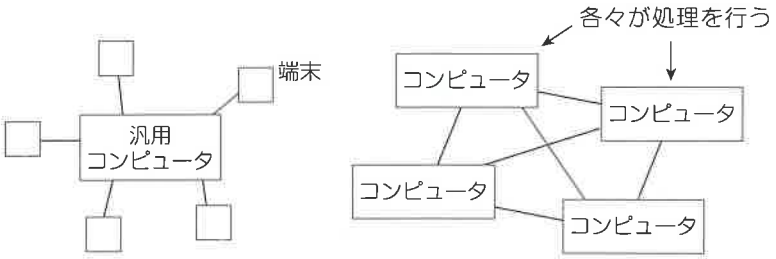
Point
・バッチ処理：ジョブを蓄積して一括処理
・オンライントランザクション処理：ジョブが発生するたびに処理
・リアルタイム制御処理：微小な時間ごとの状況変化を監視して即座に対応

②　集中システムと分散システムの比較

集中システム(集中処理システム)は，センタに配置されたコンピュータがすべてを処理し，端末は処理の投入と結果の回収を行います。これに対し，**分散システム**(分散処理システム)

は複数のコンピュータを通信回線で結び，各コンピュータが連携して処理を行います。

・集中システム　　・分散システム



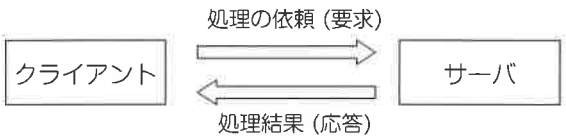
それぞれの特徴を対比させると次のようになります。

集中システム　…　・センタに障害が起こると全体に影響する
　　・全体の管理が比較的容易，セキュリティも確保しやすい
分散システム　…　・一部障害の影響を局所化できる（全体に影響が及びにくい）
　　・運用管理が煩雑になり，セキュリティ確保も困難

分散処理システムは，役割を分担する「機能分散」と，仕事の量を分担する「負荷分散」に分けられます。負荷分散において，処理要求を受け付け，適切に振り分ける装置を負荷分散装置（ロードバランサ）といいます。

③　クライアントサーバシステム

機能分散の分散処理システムの代表が**クライアントサーバシステム**です。**クライアントがサーバに処理を依頼し，サーバがそれに応える**という形で処理を進めます。



サーバには，ファイルの管理を行う**ファイルサーバ，**データベースを管理するデータベースサーバ（DBサーバ），ホームページ要求をブラウザから受け，応答を返すWebサーバなどの種類があります。

クライアントサーバ構成のうち，機能を次のような三つに分ける構成を**3層クライアントサーバシステム**とよびます。

プレゼンテーション層：データの表示や入力などのGUI部分を担当する層
ファンクション層：システムの処理機能（ビジネスロジック）を実装する層
データベース層：ファンクション層の要求によってデータベースをアクセスする層

3層クライアントサーバシステムは，Webブラウザをクライアントとした**Webシステム（Webアプリケーション）**でよく用いられます。また，3層クライアントの考えを推し進めていくと，クライアント側のマシンにはWebブラウザなどの最小限の機能しかもたせないという設計方針も考えられます。このようなクライアント端末のことを，**シンクライアント**（Thin Client）といいます。

参考：ピアツーピア

サーバとクライアントの区別を明確に設定するのではなく，各コンピュータが対等の立場で利用しあう形態をピアツーピア(peer ot peer，P2P)といいます。

④　クラウドコンピューティング

クライアントサーバの仕組みやインターネットを活用すると，手元のコンピュータの内部にデータやアプリケーションプログラムを置いておくのではなく，「データやプログラムはインターネット上に置いておき，必要なときに必要なだけ利用する」という処理形態が実現できるようになります。このような利用形態を，**クラウドコンピューティング**といいます。

クラウドコンピューティングの考えを活用したサービス事業(プロバイダ事業)には，以下のようなものがあります。

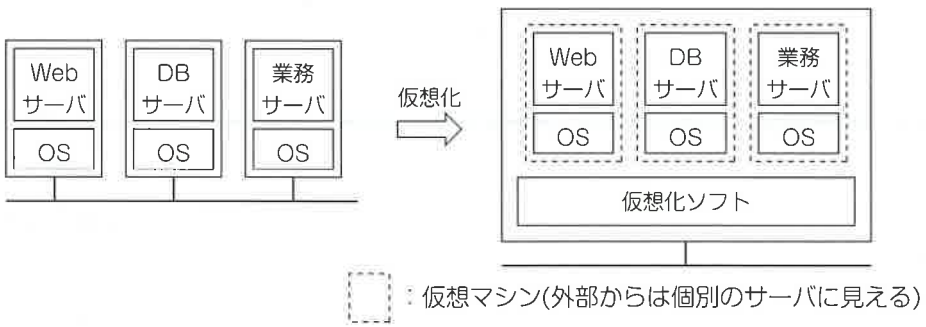
SaaS：ユーザに対してアプリケーションソフトウェアの機能を提供する

PaaS：アプリケーションはユーザが管理し，プロバイダはその開発環境を提供する

IaaS：アプリケーションや実行環境はユーザが管理し，プロバイダはハードウェアなどの基盤(インフラ)部分を提供する

⑤　仮想化技術

仮想化とは，あるコンピュータ上に，仮想マシンとよばれる「1台のコンピュータのようにふるまう環境」を作り，複数稼働させることができる技術です。仮想化技術を用いると，従来はサーバごとに別々に用意していたマシンが1台で済むなど，資源の有効活用が容易になります。また，特定の仮想マシンを別のコンピュータ上に移すなど，物理的な構成に依存しない柔軟な運用がやりやすくなります。



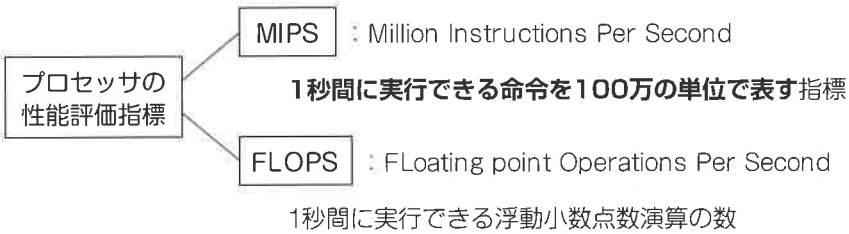
Point

- **クライアントサーバシステム：クライアントが要求を出し，サーバが応答することで処理を実現**
- **3層クライアントサーバ：機能をデータベース層，ファンクション層，プレゼンテーション層の三つに分ける**

(2)　システムの性能

①　プロセッサの性能評価指標

プロセッサ性能の指標には，**MIPS**(ミップス，ミプス)と**FLOPS**(フロップス)があります。



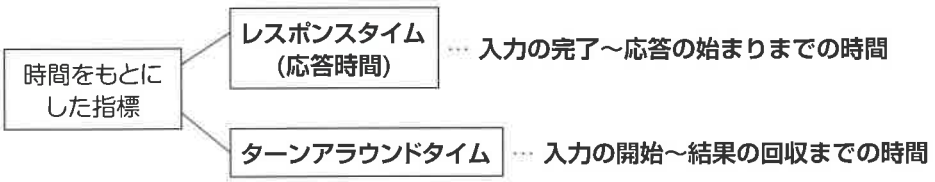
たとえば，100MIPSのプロセッサは，1秒間に$100 \times 10^6 = 100{,}000{,}000$命令を処理できます。1命令当たりの処理時間は，1/100,000,000[秒] = 10[ナノ秒]です。

②　スループット

システム性能を評価するさいに用いる用語として，**スループット**があります。これは，**単位時間にコンピュータが処理できる「仕事量」**を指す言葉です。たとえば，"20件／分"のような「単位時間当たりの処理件数」がスループットの尺度になります。

③　時間をもとにした指標

処理に要する時間をもとにした指標もあります。



④　ベンチマークテスト

ベンチマークは標準的なプログラムを実行し，システムの性能を計測・比較する方法です。MIPS値やクロック数の単純な比較だけでは難しい，異機種間の性能比較に適しています。

Point
【プロセッサの性能評価】
・MIPS値：1秒当たりの命令実行数(100万単位)
【システム性能指標】
・スループット：単位時間当たりの仕事量
・レスポンスタイム(応答時間)：入力完了から応答開始までの時間
・ターンアラウンドタイム：ジョブ提出から結果回収までの総時間

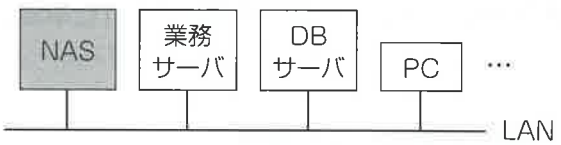
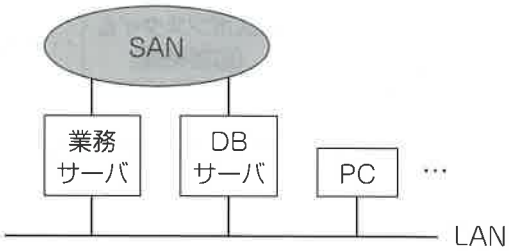
⑤　クラスタリング

クラスタリングは，ネットワークを介して複数のコンピュータを相互接続し，それ全体を「1台の仮想的な高性能・高信頼コンピュータ」としてふるまわせる技術です。負荷分散によって処理能力の向上を実現します。また，一部に障害が発生しても，その部分は切り離して他の部分で処理を引き継ぐ(フェイルオーバ)ことで，高い信頼性も実現できます。

クラスタリングなどによって非常に性能の高いシステムを構築することは，ハイパフォーマンスコンピューティング(HPC：High Performance Computing)といいます。

⑥　磁気ディスク関連のアクセス性能向上　(NAS，SAN)

コンピュータシステムでは，業務データなどを端末ごとの補助記憶装置にもたせるのではなく，ネットワーク上に大容量の専用補助記憶装置を接続して共有することがあります。このような補助記憶装置をネットワークストレージなどとよび，よく用いられるのが**NAS**，**SAN**といった仕組みです。

NAS (Network Attached Storage)	社内LANなどの通常のネットワークに直接接続し，ファイルサーバとして機能する。 
SAN (Storage Area Network)	LANとは別のストレージ専用ネットワークに，磁気ディスク装置などの記憶装置を接続する方式。 

(3)　システム信頼性

①　システムの信頼性とは

業務システムに求められる性質として，性能と並んで重視されるのが「システムの**信頼性**」です。信頼性の高いシステムほど，故障しにくく正しい動作を続けるシステムであるといえます。

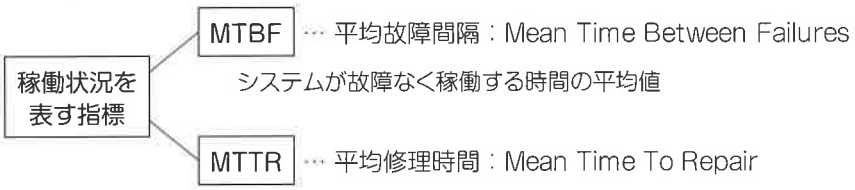
システムの信頼性を評価する概念として，(狭義の)信頼性(Reliability)，可用性(Availability)，保守性(Serviceability)の頭文字をとった“**RAS**(ラス)”と，RASに完全性(Integrity)，機密性(Security)を加えた“**RASIS**(レイシス，ラシス)”があります。

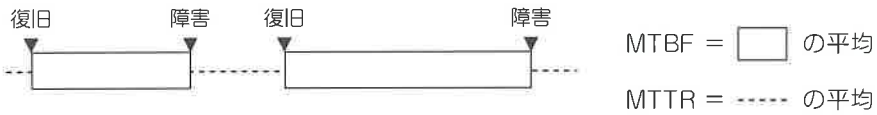
		指　標
信頼性(Reliability)	システムの故障のしにくさ	MTBF または 故障率
可用性(Availability)	システムを使用できる割合	稼働率
保守性(Serviceability)	保守の行いやすさ	MTTR
完全性(Integrity)	保全性ともいう。不整合の起こりにくさ	
機密性(Security)	犯罪や災害に対する強さ，いわゆるセキュリティ	

注：ここでいう故障率は，「故障回数／総稼働時間」で表される指標を指す。値は 1 ／ MTBF に等しくなる

②　MTBFとMTTR

システムの稼働状況を表す最も基本的な指標が，**MTBF(平均故障間隔)**と**MTTR(平均修理時間)**です。





③　稼働率

稼働率は，「全運転時間のうち，正常に稼働している(故障していない)時間の割合」を表す数値であり，可用性の指標として用いられます。稼働率が1に近いほど，可用性が高くなります。

稼働率は，MTBFとMTTRを用いて次の式で計算されます。

$$\text{稼働率} = \frac{\text{MTBF}}{\text{MTBF}+\text{MTTR}}$$

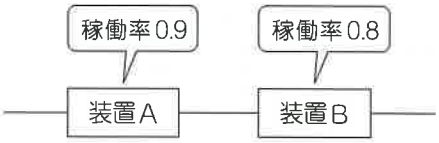
MTBFが480時間，MTTRが20時間の機器があるとき，稼働率は次のように求められます。

$$\text{稼働率} = \frac{\text{MTBF}}{\text{MTBF}+\text{MTTR}} = \frac{480}{480+20} = \frac{480}{500} = 0.96 = 96\,[\%]$$

なお，システムが「稼働していない確率」は，「1－稼働率」で求めることができます。先の例であれば，1－0.96＝0.04(4%)が「稼働していない確率」です。

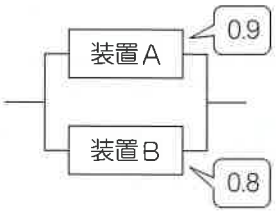
複数の装置からなるシステムの場合は，次のように考えます。

・直列に接続し，「両方が稼働していなければならない」場合の計算例



全体の稼働率
＝ Aの稼働率×Bの稼働率
＝ 0.9×0.8
＝ 0.72

・並列に接続し，「どちらか一方が稼働していればよい」場合の計算例



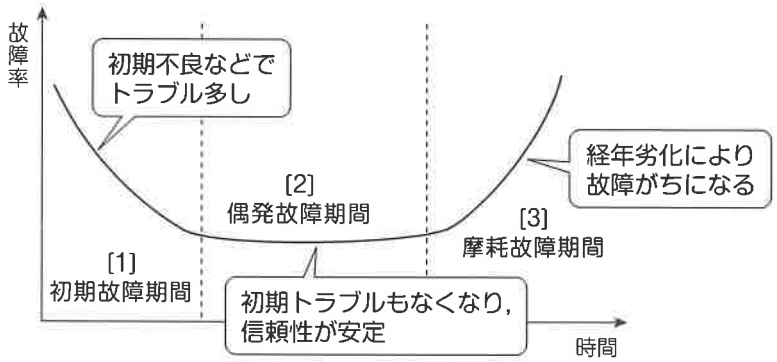
AもBも稼働していない確率

全体の稼働率
＝ 1−(1−Aの稼働率)×(1−Bの稼働率)
＝ 1−(1−0.9)×(1−0.8)
＝ 1−0.02
＝ 0.98

④　ハードウェアの保守

ハードウェア（機器）には故障がつきものであり，故障を100%なくすことはできません。

ハードウェアが故障する確率（故障率）の大きさは，時間の経過にともなって，一般に次のような推移傾向を見せます。この曲線を**バスタブ曲線**といいます。



定期点検などによって故障を未然に防ぐ**予防保守を行うことで，ハードウェアのMTBF（平均故障間隔）を長くする**ことが期待できます。

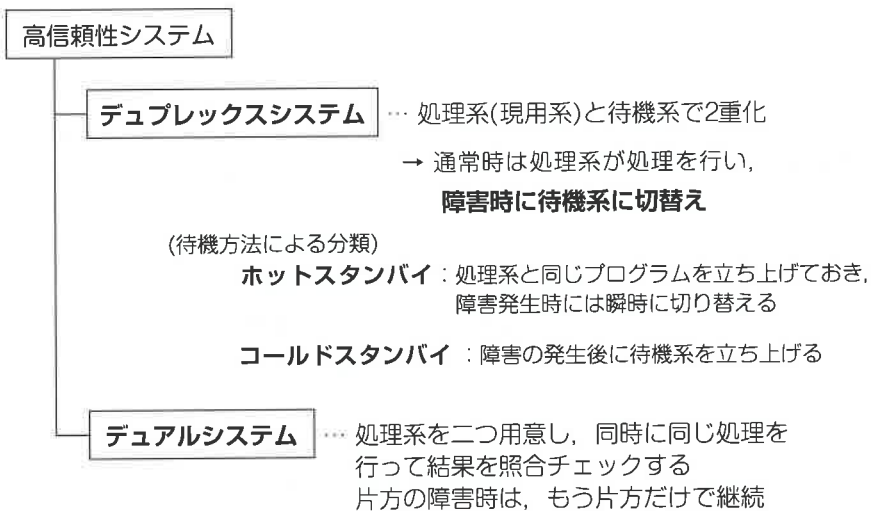
Point
- MTBF（平均故障間隔）：故障が直った直後から次の故障発生までの時間の平均
- MTTR（平均修理時間）：故障を直すのにかかる時間の平均
- 稼働率＝MTBF／(MTBF＋MTTR)

⑤　高信頼性システムの考え方

「障害が発生しても，システムの運転(稼働)を継続できるようにしておく」という概念を**フォールトトレランス**とよびます。信頼性を高めるための考え方には，フォールトトレランスを含めて，次のようなものがあります。

フォールトトレランス	障害が発生した場合でも，システムを稼働させ続けようとする考え方
フェールソフト	障害が発生した場合は故障部分を切り離すなど，機能を低下させてでも最低限の処理は継続させようとする考え方。故障部分を切り離して継続的に稼働させることを，縮退運転(フォールバック運転)とよぶ
フェールセーフ	障害による影響の拡大を防ぎ，システムをできるだけ安全な状態に導こうとする考え方。状況によっては処理の停止もやむを得ないとする
フールプルーフ	誤入力などの人為的ミスが起こりにくいように設計するという考え方
フォールトアボイダンス	システムの各構成要素の信頼性を上げるなどして，障害の発生そのものを防ぐ

フォールトトレランスを考慮したシステム構成の代表として，システムの一部や全部を2系統用意する(2重化する)**冗長構成**があります。構成方法には，次のようなものがあります。

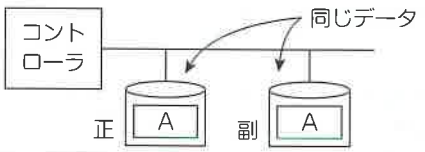
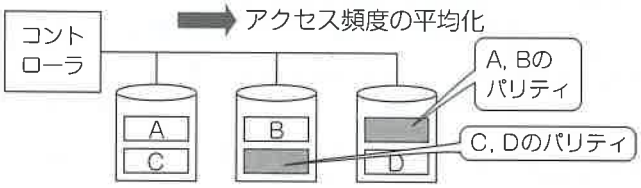


⑥　RAID

RAID(レイド)(Redundant Array of Inexpensive Disks)とは，複数台のハードディスク装置を併用する仕組みです。**ディスクアレイ**ともよばれます。主な目的は次の二つです。

- データを複数ディスクに分散して書き込む(**ストライピング**)ことで**アクセス速度の高速化**を図る
- 同じデータを複数ディスクに書き込む(**ミラーリング**)処理や障害回復用の情報(パリティビット)を加えて書き込む処理により，**信頼性の向上**を図る。

RAIDはいくつかの種類に分類できます。主なものは次のとおりです。

RAID 0	ストライピングのみ。障害復旧の機能はもたない
RAID 1	正副のディスクに同じデータを記録（ミラーリング） 
RAID 5	復旧用のパリティ情報も含め，データを分散記録する 

⑦　バックアップサイト

地震などの災害対策としては，本番システムと別の場所にバックアップ用のシステム環境を用意しておく，バックアップサイトという手法がよくとられます。バックアップサイトの種類は，ホットサイト（本番同様の環境を用意しておき，障害時には即座に切り替える）やコールドサイト（場所や基本設備だけ確保しておき，障害時にハードウェアやソフトウェアを搬入する），その中間的な位置づけのウォームサイトなどがあります。

Point

【高信頼設計の考え方】

- **フォールトトレランス：障害が起きても問題なく稼働できる性質**
- **フェールソフト：機能を落としてでも稼働を継続**
- **フェールセーフ：システムを常に安全な方向へ制御**
- **フールプルーフ：人為的ミスを予防・検知**

【高信頼システム構成】

- **デュプレックスシステム：現用（本番）系と待機系で冗長化**
- **デュアルシステム：2系列で同じ処理を行い照合**

(4)　システムの経済性

システムの導入や保守を行うときには，経済性からの評価も必要です。購入費用などのイニシャルコスト（初期コスト），消耗品費や電気代などのランニングコスト（運用コスト）の両面から，必要となる費用を見積り，費用対効果を考えながら導入・運用計画を立てます。

経済性評価によく用いられる指標としては，システムの導入から運用まで，保有に関して発生する費用の総額を表すTCO(Total Cost of Ownership)があります。

6. 2 データベース

【Guide】
関係データベースを中心としたデータベースの基本的な仕組みや，データの整合性についてどのような措置が取られるかなどについて学びましょう。

(1) データベースの基礎

① データベースとは

データベースは，データを個々のプログラム(アプリケーション)から切り離して，一元管理したものです。いわば，データの倉庫です。一元管理により，以下のような効果が得られます。

- 独立性の確保(データがアプリケーションに依存しない)
- データ内容に矛盾が発生することの防止
- データの機密保持，標準化

データベースにはいくつかの種類があり，現在では2次元の表形式でデータ内容や関連を表現する**関係データベース**が主流になっています。

② E-R図

関係データベースの設計では，**E-R図**(ERD：Entity-Relationship Diagram)がよく用いられます。E-R図では，対象となるシステムや業務で用いるデータを，**エンティティ(実体)**と**リレーションシップ(関連)**に分けて表現します。

具体的には，エンティティは四角形で，リレーションシップは線や矢線で表されます。エンティティは利用されるデータそのものに該当し，リレーションシップはエンティティ間に存在する関係に該当します。

社員は部署に所属する，という状態を表す

上の例における矢線“←”は，“社員”側に矢印があることによって，「一つの部署から見た場合，複数の社員が所属する」ということを表しています。このような数的な対応関係のことを多重度といいます。逆に“部署”側には矢印がついていないので，「1人の社員から見て，自分に関連する部署はただ一つである」ことを意味しています。

多重度は矢線を用いる代わりに，数値や記号で表記することもあります。その場合，多(任意の数が対応する)は“*”で表現されることが多くなります。

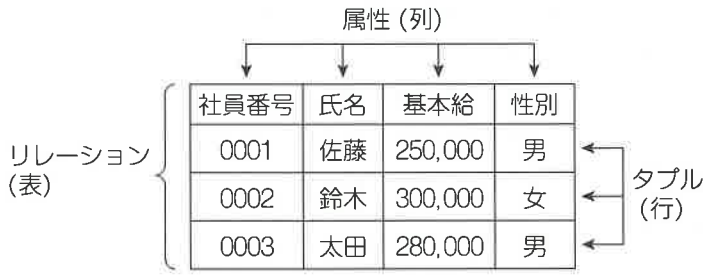
③　データベース言語

データベースの構築や操作に用いる言語を総称して，データベース言語といいます。現在主に用いられているのは，関係データベース用の**SQL**とよばれる言語です。

> **Point**
> E-R図　…　実体(エンティティ)と実体間の関連でモデル化
> SQL　…　関係データベース用の言語

(2)　関係データベースの基礎

関係データベースの表は，次のような要素から構成されており，各行が1件のデータ(レコード)に，各列が各レコードに含まれるデータ項目(フィールド)に該当します。



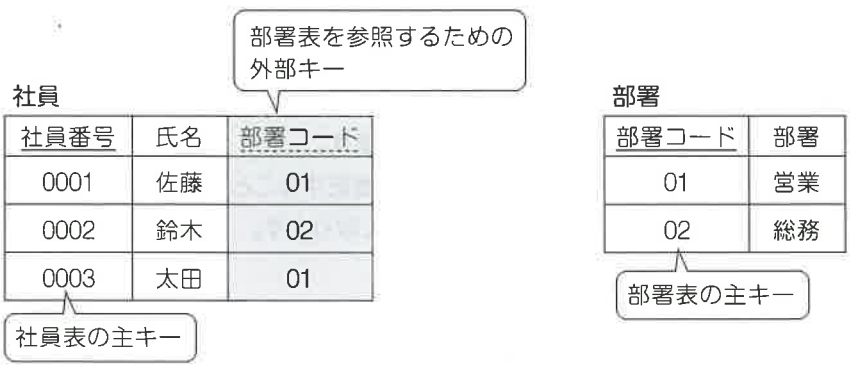
社員番号	氏名	基本給	性別
0001	佐藤	250,000	男
0002	鈴木	300,000	女
0003	太田	280,000	男

①　主キーと外部キー

ある表において「特定の行を1行だけ指定する」ために用いられる列を，**主キー**(**Primary Key**)とよびます。主キーは一つの列とは限らず，「列の組合せ」が主キーになる場合もあります。主キーに含まれる列のことを**キー項目**，含まれない列のことを**非キー項目**とよびます。

主キー部分には，「値が空値(ナル値，何もないこと)であってはならない」という非ナル制約や，「一つの表の中で値が一意でなければならない(重複してはならない)」という一意性制約が課せられます。

複数の表を扱う場合，ある表を**別の表に関連付ける**ための列を設けることがあります。これを**外部キー**といいます。外部キーの部分を用いて相手の表を参照することで，対応する情報が得られます。



参考：参照制約

データベースの整合性を保つため，外部キーには「外部キーの値が，参照先の表に必ず存在すること」という制約が課せられます。これを参照制約（外部キー制約）といいます。参照制約によって，行の追加や削除が制限を受けることがあります。たとえば，前述の図の例で“部署”表に存在しない部署コードをもつ社員の行を“社員”表にそのまま追加することはできません。

Point

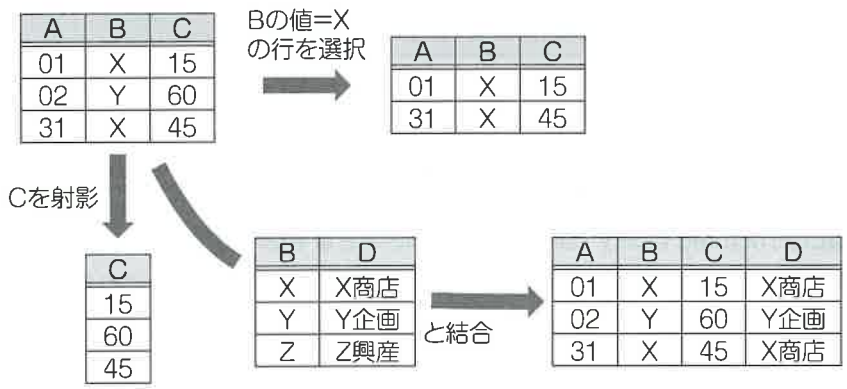
・主キー：各行を一意に特定するための属性集合（非ナル制約 ＋ 一意性制約）

・外部キー：他の表を参照するための属性（参照制約）

② 関係演算

関係演算は関係データベース特有の演算で，**選択**演算，**射影**（しゃえい）演算，**結合**演算があります。

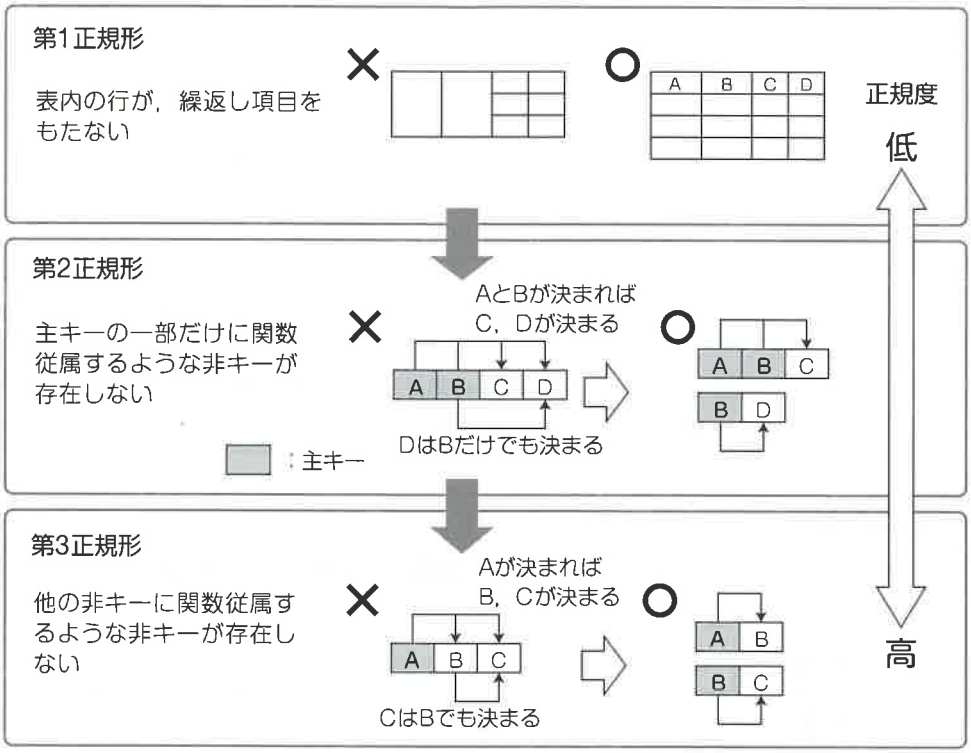
選択	表の中から条件に合致する行のみを抽出する。
射影	表の中から指定した列のみを抽出する。
結合	複数の表を条件に従って結び付け，一つの新たな表を作り出す。



(3) 正規化

冗長な（無駄，重複が多い）表には，さまざまな不整合（矛盾）が生じ，業務に深刻な影響を与えます。これを防止するために，**正規化**とよばれる「冗長性を排除する」作業が行われます。正規化の度合いはいくつかの正規形に分類されて定義されており，一般に扱われるのは第１～第３正規形です。

第２正規形からは，**関数従属性**という性質が考慮されます。これは，「Xの値が定まればYの値が一意に定まる」という関係のことです。また，第１から第２へ，第２から第３へと正規化を進めると，表がいくつかに分割されることになります。



Point
関係演算　…　選択，射影，結合
正規化　…　冗長性を排除し，更新時の矛盾を防ぐ

(4)　SQL

①　問合せ(SELECT文)

SQL(Structured Query Language)ではさまざまな処理が可能です。そのうち，「必要な情報を取り出す」ことを**問合せ**(**クエリ**)とよび，**SELECT文**が用いられます。

SELECT文の最も基本的な形は，一つの表から条件に合致する行のデータを取り出すというものです。一番基本的な書式は次のようになります。

SELECT文の書式

```
SELECT <列名リスト>        ←射影する列を指定する
    FROM <表名リスト>      ←問合せ対象の表を指定する
    WHERE <条件式>         ←検索(探索)条件を指定する
```

これにより，**FROM句**で指定した表の中から，**WHERE句**で指定した探索条件を満たす行のみが選択され，さらにそのうちSELECTに指定した列のみが射影されます。

WHEREを省略すると，すべての行が選択対象となります。また，SELECTで“*”と指定すると，全列が射影されます。

例
```
SELECT 氏名, 基本給
    FROM 社員
    WHERE 基本給 < 250000
```
基本給が25万円未満である社員の氏名と基本給を抜き出す

社員番号	氏名	基本給	部署コード
0001	鈴木	200,000	SYS1
0002	佐藤	250,000	SYS1
0003	太田	220,000	MNG1
0004	田中	280,000	MNG1

選択 ⇨

社員番号	氏名	基本給	部署コード
0001	鈴木	200,000	SYS1
0003	太田	220,000	MNG1

⇩ 射影

問合せ結果

氏名	基本給
鈴木	200,000
太田	220,000

条件式は“A = B”や“A >= B”などのように指定します。等しくないことは“<>”で表現し，NULLであることは“IS NULL”で表現します。また，ANDやOR，NOTなどの論理演算を用いることもできます。

FROM句に複数の表名を列挙し，

```
SELECT 社員.氏名, 部署.部署名
    FROM 社員, 部署
    WHERE 社員.部署コード = 部署.部署コード
```

のように記述すると，表を結合した結果が得られます。このとき，WHERE句には結合に必要な条件を記述します。

SELECT文には，そのほかに次のような機能もあります。

GROUP BY句：列名と共に記すことで，その列の値が同じ行どうしをグループ化する。合計を求める“SUM(列名)”や行数を求める“COUNT(*)”などの関数をSELECTに記述することで，グループごとの集計結果を求めることができる

ORDER BY句：列名と共に記すことで，結果をその列の昇順(ASC)や降順(DESC)に整列する

Point

・SELECT文の基本構造

```
SELECT <列名リスト>     ←射影する項目
    FROM <表名>         ←使用する表
    WHERE <条件式>      ←選択と結合の条件
```

・グループ化にはGROUP BY句，整列にはORDER BY句

② 表の定義

CREATE TABLE文を用いると，表(テーブル)を定義することができます。CREATE TABLE文の中には，各列の列名やデータ型の定義とともに，

・NOT NULL … 空値（ナル値）でない　　・UNIQUE … 値が一意である
・PRIMARY KEY … 主キーである　　・FOREIGN KEY … 外部キーである

といった，各種キー指定や制約を記述できます。

③　データ操作

行の挿入は**INSERT文**，削除は**DELETE文**，更新は**UPDATE文**で行います。また，定義した表に関して「ユーザAには表Xに対するUPDATEの権限を与える」といったように，ユーザごとの処理権限を定義するには，**GRANT文**を用います。

④　実表とビュー

記憶媒体上に実際に作成されてデータが登録される，すなわち，実際に存在する表を**実表**（テーブル）といいます。これに対し，実表からデータを抜き出したり加工したりして，ユーザが使いやすい形に作成した一時的（仮想的）な表を**ビュー**といい，SQLでは**CREATE VIEW文**で定義します。

社員番号	氏名	基本給	性別
0001	佐藤	250,000	M
0002	鈴木	300,000	F
0003	太田	280,000	M

実表

⇨

ユーザの要求
・基本給30万以上の行が必要
・性別は不要

社員番号	氏名	基本給
0002	鈴木	300,000

ビュー

⑤　ストアドプロシージャ

ストアドプロシージャは，いくつかのSQL文などをまとめた手続き（プロシージャ）をデータベースサーバに登録しておき，これを呼び出すことで実行する方式です。ストアドプロシージャは，複数のアプリケーションから共有して呼び出すことができるので，**ネットワークの通信量を軽減することができる**，などの利点があります。

(5)　DBMS

DBMS（データベース管理システム）とは，ユーザとデータベースの間に介在し，データベースの利用を支援するシステムの総称です。

①　トランザクションとACID特性

DBMSでは，多くの場合**トランザクション**という単位で処理を行います。トランザクションは，関連する**いくつかの処理をまとめた一連の処理単位**です。たとえば，「口座1から口座2へ現金を振り替える処理」などは一つのトランザクションに該当します。

トランザクションは，次に示す**ACID（アシッド）特性**という性質が保証されなければなりません。これらを保証するために，コミットメント制御などが行われます。