

平成 30 年度秋期 応用情報技術者試験 午後問 6「入室管理システムの設計」初心者向け徹底復習レポート

1. データベース統合の背景とシステム要求の構造化

複数の社内システムで運用されている関係データベース管理システム（RDBMS）を 1 台の物理サーバへ統合する際、単にハードウェアを集約するだけではなく、運用の効率化（リアルタイム性の確保）とライセンスコスト削減の両立、そして高度なセキュリティ設計の確立が重要となります。統合環境において、各システムが必要とするデータへ安全かつ円滑にアクセスできる構造を整備することが、システムアーキテクトに求められる重要な役割です。

H 社におけるデータベース統合のビジネス的背景

中堅食品会社である H 社では、従来、社内システムごとに個別サーバを用意して DB を運用していました。しかし、RDBMS のライセンスコストおよび運用コストを削減する目的から、1 台の物理サーバへの統合を決定しました。

統合にあたっては、システム間の独立性を維持しつつリソースを共有するため、統合サーバの RDBMS 上で「スキーマ」を分離して管理する方針が採られています。

システムアーキテクトの視点：単一 DB インスタンスにおけるスキーマ分離の意義

単一の DB インスタンス上でスキーマを分離する構成は、マルチテナント環境や複数システム統合における標準的なアーキテクチャパターンです。物理インフラや DB ライセンスを集約してコスト削減を図りつつ、論理的なセキュリティ境界や所有権（DBA 権限）をシステムごとに独立して管理できるため、経済性と安全性を高次元で両立できます。

人事情報配布方式の課題とビュー表による解決アプローチ

すべての社内システムで共通利用される社員情報（社員 ID や氏名など）の共用方式について、従来方式と統合後の新方式を比較整理した結果は以下の通りです。

比較項目	従来方式（月次バッチ配布）	統合後の新方式（ビュー表参照方式）
データ連携メカニズム	人事システムのマスタデータを月次処理で各 DB に配布	人事システムが管理する実表を参照するビュー表を作成し直接参照
リアルタイム性	最新情報の反映が翌月になるタイムラグが発生	人事システムの実表変更が即座に反映（タイムラグなし）
運用負荷	月次バッチ処理の実行・監視など運用負荷が大きい	データ配布処理が不要となり、運用負荷が大幅に軽減
アクセス範囲の制限	各システム側に全データがコピーされる	ビュー表により必要な列のみを非公開・限定公開可能

ビュー表を用いたセキュリティ保護の技術的意義と列の射影

H 社では、統合後も人事システム以外の社内システムから社員情報の実表（人事.社員）を直接参照させることはせず、「社内システムごとに必要な列だけをビュー表として公開（射影）し、そのビュー表を参照する方式」を採用しました。

人事システムの実表 社員 には、業務上極めて機密性の高い以下の列が含まれています（表 1 より）。

- **人事.社員 の列構成:** 社員 ID, 氏名, 勤務区分, 入社年月日, 生年月日, 社内メールアドレス, 社内電話番号, 自宅住所, 自宅電話番号, 役職, 所属組織 ID

仮に入室管理システムなどの他システムへ実表への直接参照を許可してしまうと、業務上不要な個人情報（自宅住所、自宅電話番号、生年月日 など）まで参照可能となり、情報漏洩や不正アクセスのリスクが飛躍的に高まります。

そこで本システムでは、入室管理システムに必要な最小限の列（社員 ID, 氏名, 勤務区分）のみを抽出（射影）したビュー表 HR.入室管理用社員 を公開しています（表 2 より）。これにより、機密性の高い列を物理的・論理的に遮断し、セキュリティの基本原則である「最小権限の原則（Principle of Least Privilege）」を確実に担保できる点に深い技術的意義があります。

データ構造の全体像とセキュリティアプローチを理解したところで、次章では本システムを支えるデータモデル（E-R 図および主キー構成）の解説を進めていきましょう。

2. データモデル解説：E-R 図のリレーションシップと主キーの特定（設問 1・設問 2）

データベース設計においては、実体（エンティティ）間の相互関係（カーディナリティ）を正確に定義し、レコードの変異性や一意性制約（主キー）を正しく見極めることが、整合性の高いシステムを構築するための不可欠な基礎となります。

設問 1 の解説：エンティティ間のカーディナリティ判定プロセス

本問では、表 2（主な表）および表 3（ユースケース）を基に、以下の 4 つのエンティティ間のリレーションシップ（1 対多の関係）を特定します。

1. 入室管理用社員 と 入室許可 の関係

- 1 人の社員は、異なる室や異なる期間に対して複数の入室許可を持つことができます（ユースケース「入室申請」より）。
- 逆に、1 つの入室許可レコードが対象とする社員は 1 人です。
- したがって、対応関係は「入室管理用社員」から「入室許可」へ向かう 1 対多（入室管理用社員 → 入室許可）となります。

2. 入室管理用社員 と 入退室ログ の関係

- 1 人の社員は、日常の業務の中で何回もドアを通過し、複数の入退室ログを生成します（ユースケース「入室」より）。
- 1 つの入退室ログレコードに対応する社員は 1 人です。
- したがって、対応関係は「入室管理用社員」から「入退室ログ」へ向かう 1 対多（入室管理用社員 → 入退室ログ）となります。

3. 室 と 入室許可 の関係

- 1 つの室に対しては、複数の社員に対する入室許可が出されます。
- 1 つの入室許可レコードが対象とする室は 1 つです。
- したがって、対応関係は「室」から「入室許可」へ向かう 1 対多（室 → 入室許可）となります。

4. 室 と 入退室ログ の関係

- 1 つの室のドアでは、多数の社員の入退室に伴う複数のログが記録されます。
- 1 つの入退室ログレコードに対応する室は 1 つです。
- したがって、対応関係は「室」から「入退室ログ」へ向かう 1 対多（室 → 入退室ログ）となります。

IPA 試験における E-R 図の記法ルール

IPA の午後問題における E-R 図記法（図 1 凡例）では、1 対多のリレーションシップを「1」側のエンティティから「多」側のエンティティへ向けた単方向の有向矢印（1 ――> 多）で表現します。矢印の先端（矢頭）が指し示す方向が「多」側になる点をしっかり押さえておきましょう。

なお、入室管理用社員 と 室 の間には共通属性がないため直接のリレーションシップはなく、入室許可 と 入退室ログ の関係もアプリケーションロジック側で制御されるため直接のデータ結合線は引きません。

設問 2 の解説：実表 入室許可 における主キー構成の検証

実表 入室許可 の列構成は「社員 ID、室 ID、入室許可開始年月日、入室許可終了年月日」です。この表の主キーを決定する論理は以下の通りです。

- **社員 ID と 室 ID の組み合わせだけでは不十分な理由：**同一の社員が同一の室に対して、時期を変えて（例：上期と下期など）複数回許可を得ることが想定されます。そのため、社員 ID と 室 ID だけではレコードが一意に特定できません。
- **入室許可開始年月日 を主キーに含める理由：**「いつからの許可か」という開始年月日をキー項目に加えることで、「誰が」「どの室に」「いつから」許可されたかというレコードを一意に識別できるようになります。
- **入室許可終了年月日 が主キーに含まれない理由：**表 3 のユースケース「入室許可登録」には、「すでに実表“入室許可”に同じ社員 ID、室 ID、入室許可開始年月日の行が存在する場合は、入室許可終了年月日を更新する」と明記されています。主キーの値はレコードの不変性を保つべきであり、更新（UPDATE）対象となる列は主キーから除外されるのがリレーショナルデータベースの鉄則です。

反証考察（カウンター・エクザンプル）：もし 入室許可終了年月日 を主キーに含めてしまったら？

仮に 入室許可終了年月日 を主キーに含めてしまうと、許可期間の延長などで終了年月日を更新しようとした際、主キーの値そのものを変更することになります。RDBMS の仕様上、主キーの更新は拒否されるか、または既存行の更新ではなく「別の新しい行の挿入（INSERT）」として処理されてしまいます。その結果、同じ開始年月日の許可データが重複して複数存在することになり、データの整合性が破壊され、入室可否チェックロジックが誤動作を起こす原因となります。

以上の検証から、実表 入室許可 の主キーは **社員 ID、室 ID、入室許可開始年月日**（複合キー）となります。

各表の構成要素と主キーの整理

本システムにおける表の属性構造および主キーの関係を以下の表にまとめます。

表名	種別	構成属性（列名）	主キー構成要素	備考
入室管理用社員	ビュー表	社員 ID, 氏名, 勤務区分	社員 ID	人事システムの実表 社員 から必要列のみ射影
室	実表	室 ID, 室名	室 ID	部屋の一意識別
入室許可	実表	社員 ID, 室 ID, 入室許可開始年月日, 入室許可終了年月日	社員 ID, 室 ID, 入室許可開始年月日	終了年月日は更新（UPDATE）対象列のためキー外
入退室ログ	実表	社員 ID, 室 ID, 日時, 入退室区	社員 ID, 室 ID, 日	日時により 1 回ごとのアクセスを

グ		分, 許可区分	時	一意化
---	--	---------	---	-----

基礎となるデータ構造と主キーの設計を整理できたので、次にこのデータモデル上で動作するアクセス制御の SQL クエリ（入室可否判定）の解説を進めます。

3. SQL によるビジネスロジックの実装：入室可否判定クエリの解析（設問 3）

アプリケーションにおける「アクセス可否」といったビジネスルールは、RDBMS のクエリにおいて条件式と集計関数を正確に組み合わせることで、高速かつ簡潔に実装することができます。

設問 3 の解説：入室可否チェック SQL の動作メカニズム

ユースケース「入室」において、カードリーダーに社員証が認識された際、システムは図 2 に示す SQL 文を発行してドアの開錠可否を判定します。

```
SELECT [ a ]
FROM ROOM.入室許可
WHERE 社員 ID = :社員 ID
AND 室 ID = :室 ID
AND 入室許可開始年月日 <= :今日
AND 入室許可終了年月日 >= :今日
```

埋込み変数と条件文（WHERE 句）の機能評価

クエリ内で使用されている :社員 ID、:室 ID、:今日 は、プログラム側から渡される埋込み変数です。

WHERE 句では以下の条件を同時に満たすレコードを検索しています。

1. 社員 ID = :社員 ID : カードをかざした社員本人のデータであること。
2. 室 ID = :室 ID : 入室しようとしている特定の部屋のデータであること。
3. 入室許可開始年月日 <= :今日 AND 入室許可終了年月日 >= :今日 : SQL 実行日（今日）が、許可された有効期間内（開始年月日以降かつ終了年月日以前）に含まれていること。

空欄 [a] に集計関数 COUNT(*) が入る論理的根拠（COUNT(*) と COUNT(列名) の決定的な違い）

問題文の仕様には、「入室可否をチェックし、否の場合は 0 を、可の場合は 1 以上を返す SQL 文」を作成することが求められています。

条件に適合する行が存在するかどうか（存在件数）を整数値（0 または 1 以上）として判定・返却するためには、行数をカウントする集計関数である COUNT(*) を使用します。

ここで指導上非常に重要なのが、なぜ COUNT(列名) ではなく COUNT(*) でなければならないのかという点です。

- COUNT(列名): 指定された列の値が NULL でない行数のみをカウントします。対象列の値に対する評価処理が発生します。
- COUNT(*): NULL の有無に関わらず、検索条件（WHERE 句）に一致するすべての「行（レコード）の存在数」を直接カウントします。

本問の要求は特定列の値を取得することではなく、「有効な許可レコードが 0 件か（否）、1 件以上あるか（可）」という存在確認そのものです。条件に一致する行数を評価する存在チェッククエリにおいては、NULL チェックのオーバーヘッドがなく、行数を確実にカウントできる COUNT(*) が標準的かつ最適な指定となります。

単一テーブルに対する検索・判定ロジックの解析に続き、次章では統合 DB 環境における複数スキーマ間のアクセス権限管理（GRANT 文）について解説します。

4. スキーマ分離環境におけるアクセス権限とセキュリティ制御（設問 4）

1 台の物理 RDBMS に複数の社内システムを統合する環境では、スキーマ単位でオブジェクトを隔離し、適切なアクセス権限（GRANT 文）を設定することで、データの可用性（AP の正常動作）と機密性（不正アクセスの防止）を高い次元で両立させることができます。

スキーマ隔離構造と権限の基本設計

本システムでは表 4 の通り、人事システム用として HR スキーマ、入室管理システム用として ROOM スキーマが定義されています。

- **HR スキーマ:** 所有者 DBA = HR_DBA, アプリケーション用 ユーザ = HR_AP
- **ROOM スキーマ:** 所有者 DBA = ROOM_DBA, アプリケーション用 ユーザ = ROOM_AP

各システムのアプリケーション（AP）は、セキュリティ上の理由から自システム用の AP ユーザ名で DB に接続します。したがって、入室管理システムの AP（ROOM_AP）が、人事システム（HR スキーマ）内に作成されたビュー表 HR.入室管理用社員 を参照するためには、明確な権限付与が必要となります。

設問 4(1)の解説：GRANT 文の文法解体と設問解析

図 4 の GRANT 文は、アクセス権限を与えるための標準的な SQL 構文に基づいています。

[b][c] ON [d] TO [e]

GRANT 文の基本構文は GRANT [権限の種類] ON [対象オブジェクト] TO [付与対象ユーザ] です。

- **[b] GRANT:** 権限を付与する SQL コマンド。
- **[c] SELECT:** 入室管理システムの AP は「参照だけできるように」と仕様にあるため、参照権限である SELECT を指定します。
- **[d] HR.入室管理用社員:** 付与対象となるオブジェクトです。指定ルール「スキーマ名.表名」に従い、HR スキーマの HR.入室管理用社員 と記述します。
- **[e] ROOM_AP:** 権限を与えたい相手（入室管理システムの AP 用ユーザ）である ROOM_AP を指定します。

これらをまとめると、完成する SQL 文は GRANT SELECT ON HR.入室管理用社員 TO ROOM_AP となります。

設問 4(2)の解説：権限付与を実行するユーザの論理特定

「ビュー表を参照する権限を付与する SQL 文を実行するユーザ名」は **HR_DBA** です。

RDBMS のアクセス権限仕様（1）において、「表（またはビュー表）のデータに対して所有者以外のユーザが参照を行うためには、その表の所有者（DBA）からアクセス権限を付与してもらう必要がある」と定められています。ビュー表 HR.入室管理用社員 は HR スキーマ内に作成されているため、このオブジェクトの所有者は人事システムの DBA である HR_DBA です。

したがって、ROOM_AP に対して参照権限を付与する GRANT 文を実行できる権限を持つユーザは、所有者である HR_DBA に限定されます。なお、RDBMS の仕様（2）により、ROOM_AP にビュー表へのアクセス権限が付与されれば、元となる実表 HR.社員 への直接のアクセス権限は不要となります。

RDBMS ユーザとアクセス権限の関係性まとめ

ユーザ名	種別	所有スキーマ / 役割	権限設定における立場
HR_DBA	DBA ユーザ	HR スキーマの所有者	HR.入室管理用社員 の所有権を持ち、GRANT 文を実行して権限を与える
HR_AP	AP ユーザ	人事システム実行用	HR スキーマ内のデータ処理を実行
ROOM_DBA	DBA ユーザ	ROOM スキーマの所有者	ROOM スキーマ内の実表（ROOM.入室許可 等）の所有者
ROOM_AP	AP ユーザ	入室管理システム実行用	HR_DBA から GRANT されたことで HR.入室管理用社員 の SELECT が可能

基本的権限の設定構造を把握したところで、続いて仕様変更（組織長確認の追加）に伴う自己結合（Self-JOIN）を用いた高度なビュー表拡張クエリの解説へと進みます。

5. 応用クエリ解析：自己結合（Self-JOIN）によるビュー表の機能拡張（設問 5）

システムの運用改定やセキュリティ強化に伴い、参照すべきデータ項目が増加した場合、同一テーブルを異なる役割で複数回参照する「自己結合（Self-JOIN）」技術を活用してビュー表を再定義する必要があります。

仕様変更の要求分析

管理者が入室申請を承認する際、「申請者が入室希望社員の組織長であること」を確認するセキュリティ運用の強化が行われました。これに伴い、ビュー表 HR.入室管理用社員 に、入室希望社員自身の情報だけでなく「その社員が所属する組織の組織長氏名」を新たに追加で出力するよう SQL 文（図 5）を変更することになりました。

設問 5 の解説：変更後ビュー表 SQL とテーブルエイリアスの役割解体

変更後のビュー表定義 SQL（図 5）は以下の通りです。

```
CREATE VIEW HR.入室管理用社員（社員 ID, 氏名, 勤務区分, 組織長氏名）AS
SELECT T1.社員 ID, T1.氏名, T1.勤務区分, T2.氏名
FROM HR.社員 T1, HR.社員 T2, HR.組織 T3
WHERE [ f ]
```

このクエリでは、3つのテーブル表現（エイリアス）が使用されています。

- **T1 (HR.社員):** 入室希望者（本人）の社員情報を保持する表。
- **T3 (HR.組織):** 入室希望者が所属する組織情報を保持する表。
- **T2 (HR.社員):** 組織長の社員情報を保持する表（HR.社員 表を自己結合）。

自己結合（Self-JOIN）のデータフロー視覚化

初心者にとって自己結合は混乱しやすいポイントです。データの繋がりを整理した以下のデータフロー図を頭に描きながら結合条件を組み立てましょう。

```
[T1: 社員表（入室希望社員）]
|（属性: T1.所属組織 ID）
|
```

▼ 【結合条件 1: T1.所属組織 ID = T3.組織 ID】

[T3: 組織表 (所属組織)]

| (属性: T3.組織長の社員 ID)

|

▼ 【結合条件 2: T3.組織長の社員 ID = T2.社員 ID】

[T2: 社員表 (組織長本人のレコード)]

|

└─> SELECT 句の T2.氏名 (「組織長氏名」として抽出)

結合条件 [f] の論理構築ステップ

1. **ステップ 1: 入室希望社員と所属組織の結合** 入室希望社員 (T1) がどの組織に所属しているかを特定するため、T1 の 所属組織 ID と 組織表 (T3) の 組織 ID を結合します。 → 条件式: T1.所属組織 ID = T3.組織 ID
2. **ステップ 2: 組織と組織長 (社員) の結合** 特定された組織 (T3) の組織長が誰であることを示す組織長の社員 ID と、組織長本人の社員情報を持つ T2 (社員表) の 社員 ID を結合します。 → 条件式: T3.組織長の社員 ID = T2.社員 ID
3. **ステップ 3: 全結合条件の連結** 上記ステップ 1 とステップ 2 の条件を AND で連結することで、SELECT 句にある T2.氏名 が「組織長の氏名」として正確に抽出されます。

したがって、[f] に入る適切な式は **T1.所属組織 ID = T3.組織 ID AND T3.組織長の社員 ID = T2.社員 ID** (または **T1.所属組織 ID = T3.組織 ID AND T2.社員 ID = T3.組織長の社員 ID**) となります。

6. まとめ: 応用情報技術者試験 データベース分野の徹底復習ポイント

本問題 (平成 30 年度秋期 午後問 6) は、データベースの理論設計 (E-R 図、主キー) から、実装 (SQL クエリ、集計関数)、セキュリティ・運用設計 (ビュー表、GRANT 文、スキーマ隔離) まで、実務でも直面する重要テーマが網羅された極めて良質な題材です。

復習におけるコア知識の要約

1. **ビュー表によるセキュリティ保護と射影 (Projection)** 実表 (人事.社員) の全列を公開せず、他システムに必要な最小限の列 (社員 ID, 氏名, 勤務区分) のみをビュー表として抽出・公開することで、不要な個人情報 (自宅住所, 生年月日 等) へのアクセスを防止できる。
2. **E-R 図のカーディナリティと主キー設計** エンティティ間の対応関係 (1 対多) をユースケースから正しく読み解く。IPA 記法では矢印の先端が「多」側を指す (1 —> 多)。主キー特定では一意性を確保するために必要な最小限の複合キーを考慮し、UPDATE 等で変更される列 (入室許可終了年月日 等) は主キーから除外する。
3. **SQL 集計関数による論理判定** 条件一致件数を求める COUNT(*) を使うことで、該当データの有無 (0 件なら「否」、1 件以上なら「可」) を効率よく判定できる。存在判定では NULL チェックを行わない COUNT(*) が最適。
4. **GRANT 文とオブジェクト所有権** 複数スキーマ環境において、アクセス権限 (GRANT SELECT ON ... TO ...) を付与できるのは、そのオブジェクトを所有するスキーマの DBA (HR_DBA) である。
5. **自己結合 (Self-JOIN) の活用** 「社員 (T1)」と「その組織長 (T2)」のように、同一の概念・

テーブル内に存在する関連性をたどる場合、テーブルに別名（T1, T2）を付けて組織表（T3）を介して自己結合することで目的の属性（T2.氏名）を取得する。

設問解答一覧（最終チェックリスト）

自学習の最終確認用として、本問の設問に対する模範解答を以下の一覧表にまとめます。

設問番号	設問のテーマ	模範解答・要点
設問 1	E-R 図の関連記入	・ 入室管理用社員 → 入室許可 ・ 入室管理用社員 → 入退室ログ ・ 室 → 入室許可 ・ 室 → 入退室ログ (※すべて「1 対多」の単方向矢印)
設問 2	実表「入室許可」の主キー	社員 ID, 室 ID, 入室許可開始年月日
設問 3	入室可否チェック SQL [a]	COUNT(*)
設問 4 (1)	権限付与 SQL [b] ～ [e]	[b] GRANT [c] SELECT [d] HR.入室管理用社員 [e] ROOM_AP
設問 4 (2)	権限付与実行ユーザ名	HR_DBA
設問 5	変更ビュー表の結合条件 [f]	T1.所属組織 ID = T3.組織 ID AND T3.組織長の社員 ID = T2.社員 ID