

## 問020036解説

### ◆解答

- 設問1 a イ b イ  
設問2 c イ  
設問3 d ア  
設問4 e エ f ア

### ◆解説

選手情報を管理する関係データベースの設計と運用に関する問題である。

#### 正規形と外部キー

##### ① 非正規形

属性の値が集合値や複合値で表されるレコードは非正規形である。1レコードの1データ項目に2個以上の値が存在する場合である。

##### ② 第1正規形

表において、どの属性の値を取っても、繰り返しなどの集合値や複合値を持たない表は第1正規形である。

##### ③ 第2正規形

第1正規形の表の属性の値には、主キーが決まると一意に決まる値と決まらない値が存在する。全ての非キーの属性値を、主キーが決まると一意に決まる属性値と一意に決まらない属性値に分離し、2つのレコードを設定する場合、これらのレコードは第2正規形である。

##### ④ 第3正規形

第2正規形において、非キー以外の属性で主キーとなり得る属性とその主キーが決まると一意に決まる属性を分離して新しいレコードを設定し、元のレコードから新しいレコードの属性を除いた属性の集まりと新しいレコードの主キーの属性を元のレコードの外部キーとして加えてレコードを設定する。これを第3正規形という。

##### ⑤ 外部キー

外部キーとは、その属性を主キーとするレコードが、外部キーが属するレコード外に存在する場合で、外部に存在するレコードを参照できる場合である。

#### ORDER BY を使用したSQLの基本文

日付順に受注情報を表示する。

```
SELECT * FROM 受注表 ORDER BY 受注日
```

「どの表から」、「結果を昇順で出力する」の構文である。ORDER BY文節は分類順序を表す。文節にASCを指示すると昇順、DESCを指示すると降順を表し、省略するとASCとなる。

ORDER BY 列名で分類の特定列を指定する。特定の複数列で分類することも可能である。複数列で分類する場合、ORDER BY 列名, ..., 列名の形式になる。最初の列名が分類の第1キーとなり、順次第2キー、第3キー、...となる。更に、個々の分類キーの昇順、降順を指定する

場合、それぞれの列名の後にASC、DESCを指定し、ORDER BY 列名 ASC, 列名 DESC, 列名 ASC, …などの形式で表す。受注表は、(伝票番号、顧客番号、受注日)のレコード様式である。

並べ替えの基準となる列が表の最初の列の場合には、列名を記述する代わりに「ORDER BY 1」で表すこともできる。

受注表

伝票番号	顧客番号	受注日

### ORDER BY を使用したSQL文の検索例

単価が1000以下の商品を単価の高いもの順に表示する。

```
SELECT * FROM 商品表 WHERE 単価 <= 1000 ORDER BY 単価 DESC
```

### GROUP BY を使用したSQLの基本文

商品番号毎の受注数量の合計を表示する。

```
SELECT 商品番号, SUM(数量) FROM 受注明細表 GROUP BY 商品番号
```

GROUP BY 句はグループ化を伴う操作であり、「どの表から」、「グループ化して」、「式の結果を取り出す」の構文である。特定の列の値に従って行をグループ化する。特定の列の値が同じ行をまとめて、表をいくつかのグループに分ける場合に使用する。GROUP BYに指定できるのは、列名でSELECT文節に表れているものである。SELECT文節にグループ化する列名と集計関数がある場合、GROUP BYの指定がないとエラーになる。また、GROUP BYを指定すると、SELECT文節で指定できるものは、GROUP BYで指定した列名またはその列名による集計関数しか記述することができない。GROUP BYによってグループ化する場合、グループの中で値が一意に定まらない列名を、SELECT文節で指定することができない。ただし、GROUP BYで指定した列名をSELECT文節で必ずしも指定する必要はない。

複数の列を対象にグループ化する場合は、GROUP BY 列名, …, 列名の形式を用いる。集計関数を複数個使用する場合、次の構文になる。

```
SELECT 列名, 集計関数(列名), …, 集計関数(列名) FROM 表名 GROUP BY 列名
```

受注明細表は、(伝票番号、商品番号、顧客番号、数量)のレコード様式である。

### GROUP BY を使用したSQL文の検索例

受注表から2件以上の注文を受けている顧客の最新の受注月日を求める。

```
SELECT 顧客番号, MAX(受注日) FROM 受注表  
GROUP BY 顧客番号 HAVING COUNT(*) >= 2
```

最新の受注日を求めるためにMAX関数を利用する。HAVING文節は、GROUP BYによって、グル

ープ化されたデータに検索条件を設定し、データを絞り込むことができる。

WHEREとGROUP BY を指定すると、検索条件が与えられた後にグループ化が行われる。GROUP BYとHAVINGを使用すると、グループ化が行われた後に検索条件が与えられる。GROUP BYと集合関数を使用して値を求め、その結果を条件に従って評価する場合、条件式にHAVINGを使用する。WHEREを使用するとエラーになる。

COUNT(\*)は、行数(レコード数)のカウントである。重複を許す場合である。COUNT(DISTINCT, 列名)の場合は重複を除いた行数である。

### GROUP BY、ORDER BY を使用したSQL文の検索例

受注明細表から顧客番号と伝票番号毎の受注件数を求めて顧客番号と伝票番号順に表示する。

```
SELECT 顧客番号, 伝票番号, COUNT(*) FROM 受注明細表
      GROUP BY 顧客番号, 伝票番号 ORDER BY 顧客番号, 伝票番号
```

### 設問 1

aは、日程や試合結果の情報を選手表を利用して管理する仕組みを考えた場合の問題点を問う内容で、日程や試合結果の項目を設定すると、日程や試合結果の属性の値が集合値や複合値で表されるレコードとなり、非正規形となる。そこで第一正規化して、どの属性の値を取っても、繰り返しなどの集合値や複合値を持たない表にする。aの求める答えはイとなる。

表の項目に繰り返しが発生するのは非正規形の形であり、これを改善するのは第一正規化である。

bは、結果表のレコード件数を問う問題で、8チーム総当たりの試合数は、 ${}_8C_2 = 28$ であり、1試合2レコードであるから計56レコードとなる。求める答えはイとなる。

### 設問 2

GROUP BY、ORDER BYに関する問題である。GROUP BY句は、グループ化を伴う操作で、「どの表から」、「グループ化して」、「式の結果を取り出す」の構文である。チーム表のチーム番号、チーム表のチーム名に従って行をグループ化し、勝点の合計、得点の合計を求める。

ORDER BY句は、「どの表から」、「結果を昇順で出力する」の構文で、ORDER BYは分類順序を表す。文節にASCを指示すると昇順、DESCを指示すると降順を表し、省略するとASCとなる。ORDER BY 列名で分類の特定列を指定する。特定の複数列で分類することも可能である。複数列で分類する場合、ORDER BY 列名, ..., 列名の形式になる。第一列名を勝点、第二列名を総得点として、降順に整列する。答えは次のようになる。

```
GROUP BY チーム表. チーム番号, チーム表. チーム名
ORDER BY 勝点 DESC, 総得点 DESC
```

求める答えはイとなる。

### 設問3

```
SELECT 打席表.打者, 選手表.氏名,  AS 集計数
FROM 打席表, 選手表
WHERE 打席表.打者 = 選手表.選手番号 AND 打席表.打撃結果
      = (SELECT 打撃表.打撃結果 FROM 打撃表 WHERE 打撃表.名称 = 'ホームラン')
GROUP BY 打席表.打者, 選手表.氏名 ORDER BY 集計数 
```

このSELECT文は、打席表の打者(選手番号)、選手表の選手氏名別にホームラン数を集計し、集計数の項目に集計数の降順に表示する内容である。打席表からホームラン数を求める方法は、打席表の打者(選手番号)と選手表の選手番号が一致するレコードで、打撃表の打撃結果が003となるレコードを副照会で求め、そのレコード件数を求めればホームラン数が求まる考え方を使用している。d 1の答えはCOUNT(\*)、d 2はDESCで、求める答えはアとなる。

### 設問4

eは、シーズン途中で選手が別のチームに移籍した場合の問題点であるから、選手個人に関係する内容で、チームの情報が異なる場合に発生する問題であるから、チーム毎のホームラン数となる。移籍以前に打っていたホームランの数が移籍後のチームのホームラン数に加算される。eの求める答えはエとなる。

fは、同一選手の選手番号が異なるため、選手毎のシーズンを通したホームラン数の集計がSQL文では困難になる。fの求める答えはアとなる。

この対策として、従来は選手表で管理していたものを、所属選手表と選手表を使用して管理するようになると、選手番号はリーグで唯一の番号とし移籍前後で変化しない、選手の所属チームは所属選手表で管理できるため、e、fの問題を解消することができる。