【１】トランザクション処理　　　 　　　　　　　…テキストP.（　　　　）

１．トランザクションとは

システムで処理される、複数の処理を１つにまとめ、分離できない処理の単位。

２．トランザクションの例

Ａさんの銀行口座からＢさんの銀行口座に１万円振り込む

トランザクション

処理１

Ａさんの口座から10,000円差し引いた

残高で更新する

処理２

Ｂさんの口座に10,000円加えた

残高で更新する

※処理１と処理２の順番を変えることはできるが、どちらかだけが処理されるということはない。

データベースは不特定多数からのアクセスが可能な資源であるため、同時に複数の利用者が処理を行い、データベースの内容を変更（更新・追加・削除）すると、処理結果に矛盾が生じる場合がある。

（１）処理結果に矛盾が生じる例

特別な制御を用いない場合のデータベースの状態

①　プログラムＡがデータＸを読む。

②　プログラムＢがデータＸを読む。

③　プログラムＡでＸに50を加え、その結果をデータベースに書く。

④　プログラムＢでＸから50を引き、その結果をデータベースに書く。

①～④の処理が、この順番で行われると本来150のデータが100になってしまう。

Ｘを読む

Ｘに50を加える

Ｘの更新処理

Ｘは150

Ｘは200

プログラムＡ

データベース

Ｘ

③

①

②

④

Ｘを読む

Ｘから50を引く

Ｘは150

Ｘは100

プログラムＢ

Ｘの更新処理

150

↓

200

↓

100

（２）［①　　　　　　　　　　　］

複数の利用者が同時に処理を行う場合、一方の利用者が利用している間は、該当するデータベースやデータをロックし、他の利用者が利用できないようにする機能。ロックを解除するまでは、他の利用者は待ち状態となる。

①　プログラムＡがデータＸを読むと同時にロックをかける。ロックは他の利用者のアクセスを禁止し、処理の終了まで待たせる機能である。

②　プログラムＢはプログラムＡの終了後、データＸを読み、処理を行う。

③　両プログラムの実行後の結果は、150となる。

Ｘを読む

Ｘに50を加える

Ｘの更新処理

Ｘは150

Ｘは200

プログラムＡ

Ｘを読む

Ｘから50を引く

Ｘの更新処理

プログラムＢ

データベース

150

↓

200

Ｘ

アクセスできない

ロック

プログラムＡの処理終了後

処理終了

プログラムＡ

Ｘを読む

Ｘから50を引く

Ｘの更新処理

プログラムＢ

Ｘは200

Ｘは150

データベース

200

↓

150

Ｘ

ロックの解除

３．ロックの種類

|  |  |
| --- | --- |
| ［②　　　　　　　　　　　］ | ロック中に他の利用者はデータを参照することができる。  更新・削除・追加はできない。ロック中、他の利用者が更に共有ロックをかけることができる。 |
| ［③　　　　　　　　　　　］ | ロック中に他の利用者はすべての操作が禁止になる。  データを参照・更新・削除・追加できない。ロック中、他の利用者はロックをかけることができない。 |

（１）ロックの種類とアクセス可否の関係

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | 他の利用者の操作 | | | |
|  |  | 参照処理 | 更新処理 | 共有ロック | 専有ロック |
| ロックの種類 | 共有ロック | **○** | × | **○** | × |
| 専有ロック | × | × | × | × |

（２）［④　　　　　　　　　］

複数のトランザクションで各々がロックを行った結果、お互いに使いたいデータがロックされた状態になることによって、処理が進まなくなる現象。この状態になった場合は、一方のトランザクション処理をキャンセルする措置が取られる。

次の①～④の順で処理を行う場合

①　プログラムＡがデータＸを読むと同時にロックをかける。

②　プログラムＢがデータＹを読むと同時にロックをかける。

③　プログラムＡがデータＹを読む。

④　プログラムＢがデータＸを読む。

③

①

データベース

データＸ

データＹ

Ｘを読む

Ｙを読む

プログラムＡ

※Ｙは利用中なので

待ち状態となる

ロック

②

④

Ｙを読む

プログラムＢ

Ｘを読む

※Ｘは利用中なので

待ち状態となる

ロック

４．トランザクション処理に求められる特性

（１）［⑤　　　　　　　　　　］

データベースの状態が矛盾なく一貫した状態を保証するために、データベースが備えるべき４つの特性。

|  |  |
| --- | --- |
| ［⑥Atomicity　（　　　　　　　　）］ | トランザクションの処理結果が完了、または全く行われていない状態で終了する。中途半端な状態を認めない。 |
| ［⑦Consistency（　　　　　　　　）］ | トランザクションがどのように終了しても内容に矛盾がない状態。 |
| ［⑧Isolation　 （　　　　　　　　）］ | トランザクションを同時に実行した場合と順番に実行した場合の処理結果が同じになる。 |
| ［⑨Durability （　　　　　　　　）］ | トランザクションが正常に終了したら、障害が発生しても結果が消失しない。 |

≪範例１≫

DBMSの排他制御に関する記述のうち、適切なものはどれか。

ア　アクセス頻度の高いデータの処理速度を上げるためには、排他制御が必要である。

イ　処理速度を上げるため、排他制御を行うデータの範囲は極力大きくすべきである。

ウ　データアクセス時のデッドロックを防止するために、排他制御が必要である。

エ　複数の人が同時に更新する可能性のあるデータには、排他制御が必要である。

≪解答≫　エ

排他制御は、ある利用者がデータを使用している間は、当該データを他の利用者が使用することを禁止する制御です。なお、データを利用禁止にするためにロックするデータの範囲をロックの粒度（ロックレベル）と呼びます。ロックの粒度を大きくすれば、他の利用者の資源待ちの確率が高くなり、全体の処理の効率が低下します。排他制御では、複数の利用者が同じ資源を同時に必要とした場合、排他的に使用してしまうことで、お互いに待ち状態となり処理が停止する、デッドロックの発生に注意する必要があります。