

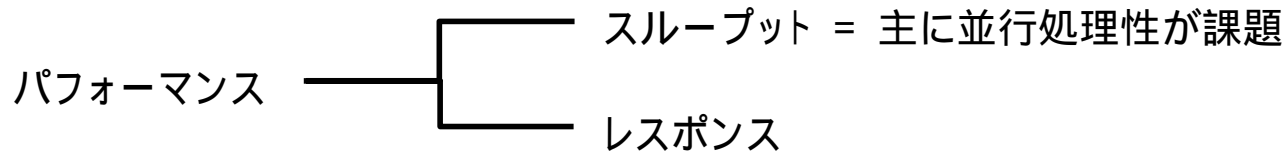
# 第3章

## DB2 アプリケーション開発の基礎

### 3. DB2 アプリケーション開発の基礎

---

- この章ではパフォーマンス・チューニングに観点を置いた開発の方法についてご紹介します。



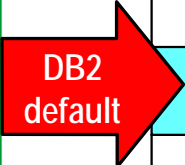
- 3.1. 分離レベル (Isolation Level) の指定
- 3.2. より良い SQL 文の書き方
- 3.3. 開発を助けるツール
- 3.4. 開発ヒント集

# 3.1. 分離レベル (Isolation Level) の指定

- 分離レベルとは

- マルチ・ユーザー環境でデータを共有するために、アプリケーション要件に応じてデータアクセスを制御し、望ましくない現象を発生させないようにします。
- 分離レベルはアプリケーション設計者 / 開発者が指定・要求します。
- 望ましくない現象とは
  - 更新の紛失
  - 未コミットデータの読取り
  - 反復不能読取り
  - 幻像読取り

DB2 は要求された分離レベルを実現するために、主にロックを使用して排他制御を行います。

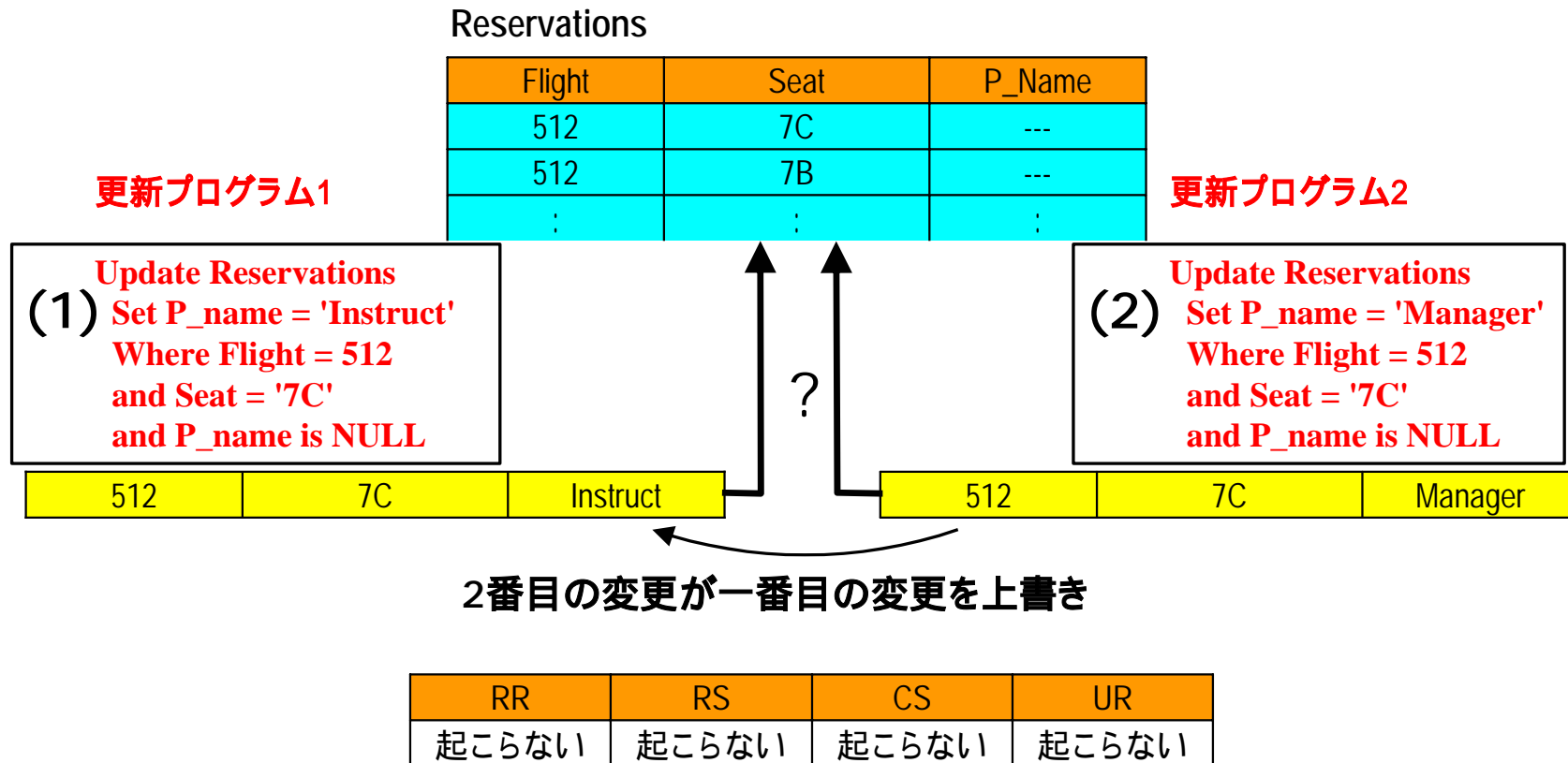


ISO 分離レベル	DB2 分離レベル	未コミットデータへのアクセス	反復不能読取り	幻像読取り
Serializable	Repeatable Read (RR)	起こらない	起こらない	起こらない
Repeatable Read	Read Stability (RS)	起こらない	起こらない	起こる
Read Committed	Cursor Stability (CS)	起こらない	起こる	起こる
Read Uncommitted	Uncommitted Read (UR)	起こる	起こる	起こる

↑ 整合性が高い  
↓ 並行性が高い

# 更新の紛失 (Lost Updates)

- ある更新処理による更新データが確定される前に、他の更新処理により変更されてしまう。



# 未コミットデータへのアクセス (Dirty Reads)

- コミット前のデータを読み取ってしまい、更新データがロールバックされた場合には、読取りデータは実際とは異なるデータとなる。

Reservations

Flight	Seat	P_Name
512	7C	Instruct
512	7B	Manager
⋮	⋮	⋮

更新プログラム

参照プログラム

(1) **Update Reservations**  
Set P\_name = 'NULL'  
Where Flight = 512  
and Seat = '7C'  
and P\_name = 'Instruct'

(3) **Rollback**

512	7C	---
-----	----	-----

512	7C	---
-----	----	-----

(2) **Select Seat**  
From Reservations  
Where P\_name is  
NULL

(4) **正しくない結果セット**

RR	RS	CS	UR
起こらない	起こらない	起こらない	起こる

# 反復不能読取り (Non Repeatable Reads)

- 1つの作業単位内で、同じSELECT文が実行された場合に、同じ照会結果が保証されない。

Reservations

Flight	Seat	P_Name
512	7C	---
512	7B	---
:	:	:

参照プログラム

更新プログラム

(1) **Select Seat**  
**From Reservations**  
**Where P\_name is NULL**  
**and Flight = '512'**

(3) (1) の select 文を再度実行  
すると Seat が '7C' の行は  
戻されない

512	7C	---
512	7B	---

512	7B	---
-----	----	-----

(2) **Update Reservations**  
**Set P\_name = 'Instruct'**  
**Where Flight = 512**  
**and Seat = '7C'**  
**and P\_name is NULL**

RR	RS	CS	UR
起こらない	起こらない	起こる	起こる

# 幻像読取り (Phantom Reads)

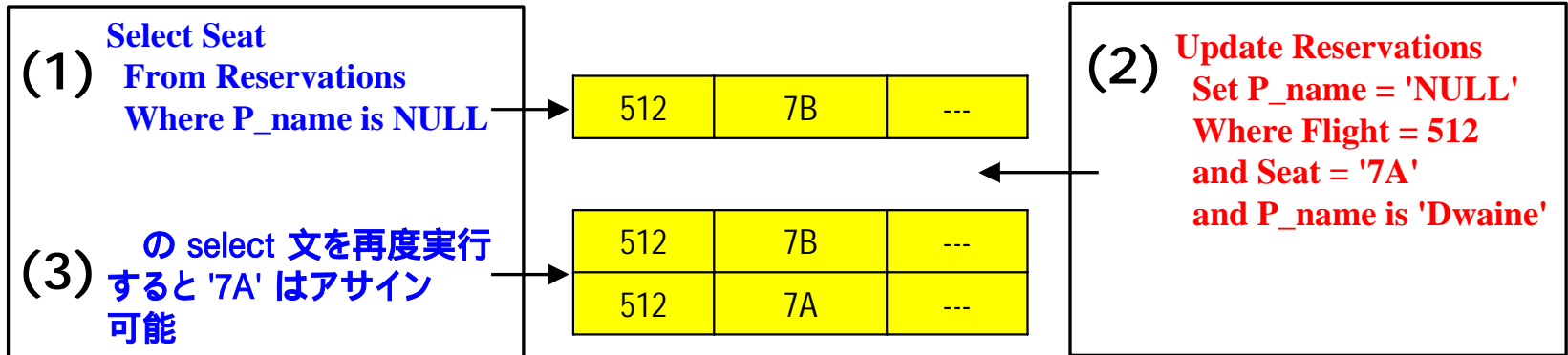
- 同一作業単位内で、同じSELECT 文を実行した場合に、他の更新処理により条件に合致するようになったデータまで照会結果に入る。

Reservations

Flight	Seat	P_Name
512	7B	---
512	7A	Dwaine
⋮	⋮	⋮

参照プログラム

更新プログラム



RR	RS	CS	UR
起こらない	起こる	起こる	起こる

# 推奨分離レベル

- 推奨分離レベルは以下のとおりです。

アプリケーションのタイプ	高度のデータ整合性が必要	並行処理によるスループットを優先する
読書きトランザクション	RS	CS
読取り専用トランザクション	RR	UR

- CS (デフォルト)が望ましい場合
  - 他の分離レベルの要件がない場合は、並行性の高いCSが推奨。
- RRが望ましい場合
  - 同一トランザクション内で同じ行を複数回アクセスし、それらが常に全く同じ内容であることを保証したい場合。
    - ・ ただし多くのロックが獲得され、並行処理性を低下させ、システム全体のパフォーマンスが低下する可能性があるため、可能な限りRRの使用は避けましょう。
- RSが望ましい場合
  - 同一トランザクション内で同じ行を複数回アクセスし、前回得られた結果を保証したい場合。
- URが望ましい場合
  - すばやく可能な行をSELECTし、アサインし、確定は後で行ってもかまわないなど、今の状況を取りあえず確認したい場合。
  - 読取り専用の表を使用する場合。

# 分離レベルの指定方法

---

- アプリケーションのタイプによる分離レベルの選択
  - アプリケーション要件により、分離レベルを選択する。
- 設定方法
  - 埋め込みSQL
    - PREP/BIND コマンドのISOLATION オプション
  - CLI、ODBC アプリケーション
    - db2cli.ini ファイルに設定する方法
  - アプリケーションで設定する方法
    - CLI
      - SQL\_SetConnectAttr() のSQL\_ATTR\_TXN\_ISOLATION オプション
      - SQL\_SetStmtAttr() のSQL\_ATTR\_STMTTXN\_ISOLATION オプション
    - Java
      - Connection クラスの setTransactionIsolation() メソッド
  - CLP (コマンド行プロセッサ)
    - CHANGE ISOLATION TO [CS |RR |RS |UR |NC ]
  - SQL 文にWITH 指定
    - SELECT \* FROM TBL WITH UR

# バインド時の指定方法

- バインドするアプリケーションが実行する SQL の分離レベルとなります。
- C の場合 (CLI を除く) は PREP/BIND を実行するときの isolation オプションで分離レベルを指定
  - 何も指定しないとデフォルトの CS になります。
  - BIND 時に指定しない場合は PREP で指定した分離レベルが使用されます。
- CLI/Java の場合は db2cli.lst ファイルをバインドするタイミングで指定
  - 最初に一回だけ実行する必要があります。
  - 一回も実行していないと、デフォルトの CS で自動的にバインドされます。

## bind 時に RR を指定する例

```
> bind bindfilename isolation rr
```

ここで rs や ur を指定する

# db2cli.ini での指定方法

- データベース単位で設定します
  - そのデータベースに接続するアプリケーションが実行する SQL の分離レベルになります。
- CLI アプリケーションと Java アプリケーションに対して使用可能
  - db2cli.ini ファイルの中でキーワード TXNISOLATION を設定します。
    - 1 = UR:非コミット読み取り
    - 2 = CS:カーソル固定
    - 4 = RS:読み取り固定
    - 8 = RR:反復可能読み取り
  - 変更後は次にアプリケーションを稼動したタイミングで反映されます。

## db2cli.ini で RS を設定する例

```
[SAMPLE]  
TXNISOLATION=4
```

SAMPLE データベースに対して設定

# Java プログラムの中での指定方法

- 接続 (Connection) 単位で設定します
  - 同一接続を使用するすべての SQL に対して有効となります。
- 設定するタイミング
  - SQL 文を指定する前に設定します。

## Java プログラムの中で UR を設定する例

```
Connection con;  
con.setTransactionIsolation(con.TRANSACTION_READ_UNCOMMITTED)  
ResultSet rs = con.createStatement("select id from staff where name='John'");  
rs.execute();
```

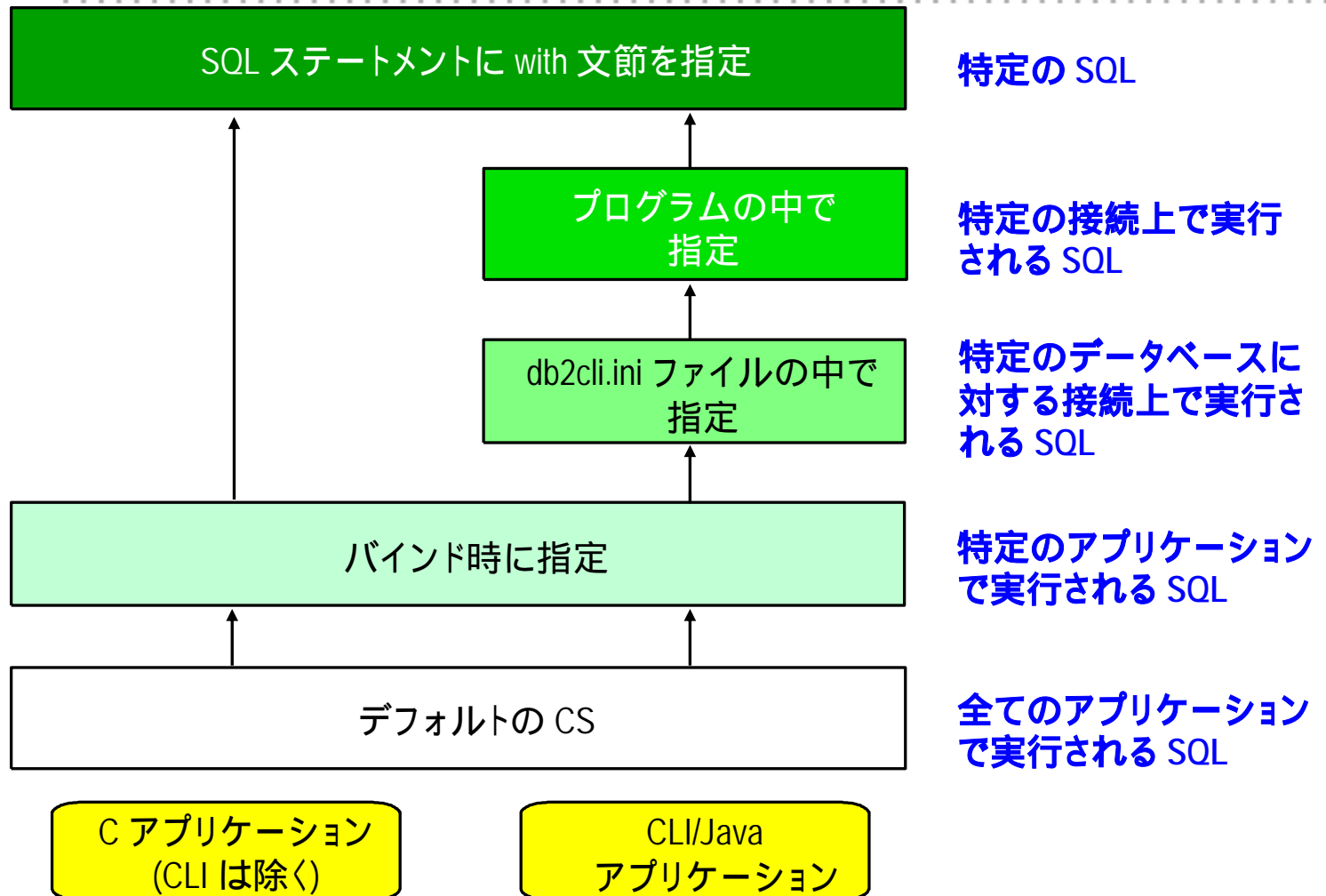
Java分離レベル	DB2分離レベル
TRANSACTION_SERIALIZABLE	Repeatable Read (RR)
TRANSACTION_REPEATABLE_READ	Read Stability (RS)
TRANSACTION_READ_COMMITTED	Cursor Stability (CS)
TRANSACTION_READ_UNCOMMITTED	Uncommitted Read (UR)



# SQL ステートメントに with 分離レベルを指定する方法

- 指定できるのは以下の SQL ステートメントです。
  - SELECT
  - INSERT
  - UPDATE
  - DECLARE CURSOR
- with 文節の使用に関しては、いくつかの条件があります。
  - with 文節を副照会で使用することはできません。
  - with UR オプションは、読み取り専用の操作にのみ適用されます。このオプションが他の状況で使用された場合、ステートメントは "UR" から "CS" に自動的に変更されます。
  - ステートメントのデフォルト分離レベルは、ステートメントがバインドされるパッケージの分離レベルです。
  - ステートメント・レベルの分離レベルは、ステートメントがあるパッケージに指定された分離レベルをオーバーライドします。
  - SQL プロシージャ - では使用できますが、MacroPSM では使用することができません。

# 分離レベルの優先順位



# ロックに関する考慮点 -COMMIT-

- **トランザクションの区切りには必ず COMMIT を入れ、できるだけ早くロックを解放しましょう。**
  - 分離レベルにかかわらず、更新されたオブジェクトはCOMMIT / ROLLBACK が実行されるまでロックを保持します。
  - **RR、RS** のアプリケーションでは、照会処理でもロック (Read Lock) が保持されるため、トランザクションの区切りに必ず COMMIT を入れましょう。
  - UR のアプリケーションでも、動的 SQL の場合にはシステム・カタログ表へのロックが取得されます。
- SQL Exception の際も必ず COMMIT / ROLLBACK を行う必要があります。

# ロックに関する考慮点 - デッドロック -

---

- デッドロック

- 複数のアプリケーションがお互いにロックしているデータを待機してしまう状態です。

- デッドロックを起こさないためには

- COMMIT を適正な頻度で実行する。
- 適正な分離レベルを選択する。
- 可能であれば RR をやめる。
- 可能であれば FOR UPDATE を指定する。
- 適正なインデックスを使用する。

## 3.2.より良い SQL 文の書き方

---

- 一般的な方法 (標準 SQL の範囲で対応できること)
  - 列の選択時に必要なものだけを指定する
  - 更新はトランザクションの最後に行う
  - FOR UPDATE 文節を指定する
- DB2 固有の書き方になる方法
  - FETCH FIRST n ROWS ONLY 文節を指定する

# 列の選択時に必要なものだけを指定する

---

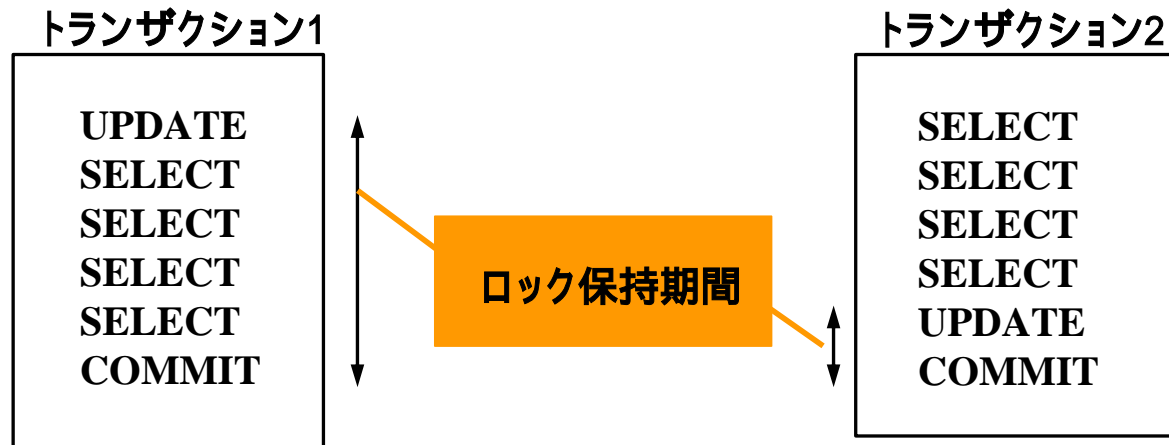
- パフォーマンス向上のために

- `select * from ...` を使用しない

- 無用な列の SELECT は、データベース・サーバーだけでなく、列データを受け取るアプリケーション側のオーバーヘッドやネットワーク・オーバーヘッドにもつながります。
    - アプリケーション要件を確認して、不要な列は SELECT 対象からはずしてください。

# 更新はトランザクションの最後に行う

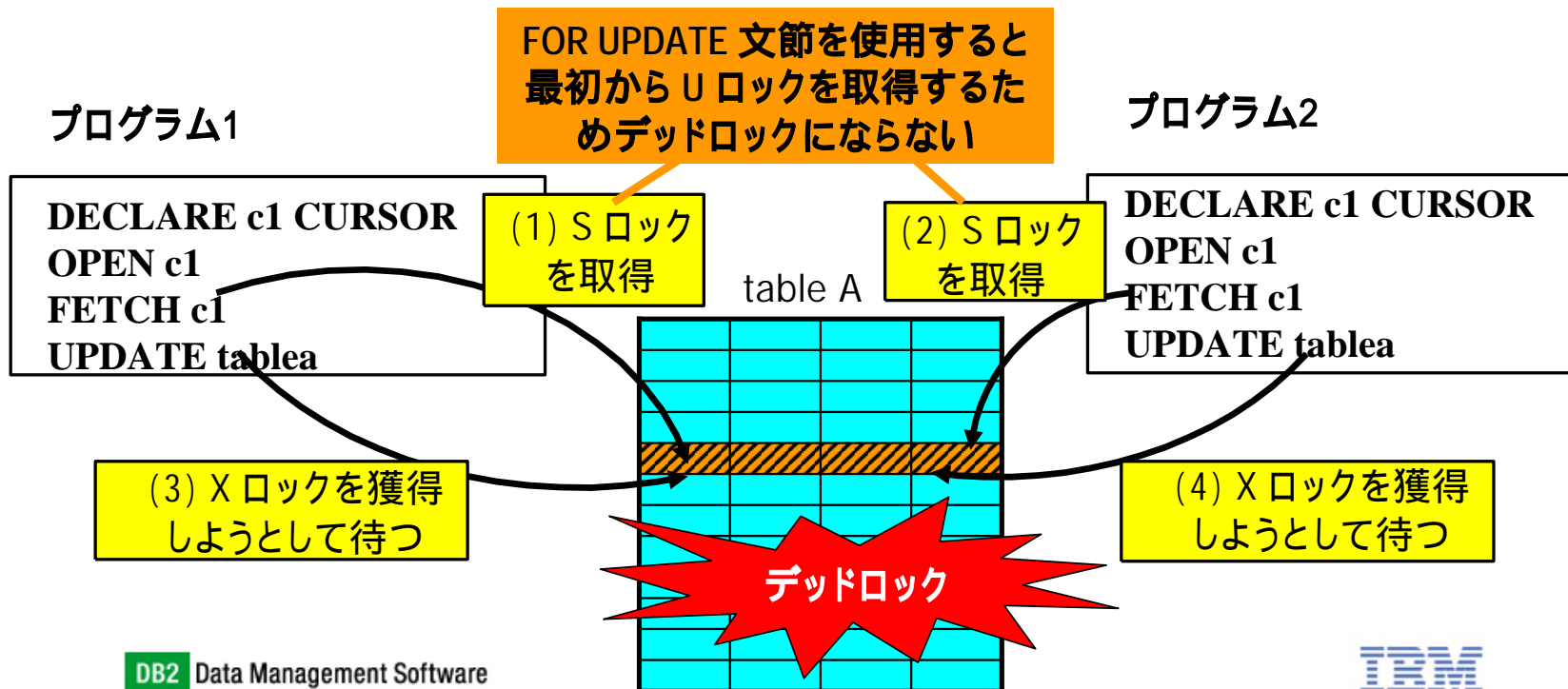
- 並行処理性を上げるために
  - 更新処理はできるだけトランザクションの最後 (COMMIT の直前) で実行するようにしてください。
    - INSERT、UPDATE、DELETE はトランザクションが終了するまで、排他ロックを取得します。
    - これらの更新処理をトランザクションの最後で実行することで、最大の並行性が得られます。



# FOR UPDATE 文節

- デッドロックの可能性を下げるために

- 同一トランザクションで取り出したデータを更新する場合
  - FOR UPDATE を指定することでデッドロックの可能性を減らします！
  - 並行処理性とのトレードオフになります。



## FOR UPDATE 文節 (C)

---

- FOR UPDATE 文節を指定してカーソルを宣言する。
- WHERE CURRENT OF カーソル名 を指定して UPDATE 文を実行する。

```
EXEC SQL DECLARE c1 CURSOR FOR select id from employee FOR UPDATE OF job;  
EXEC SQL OPEN c1;  
EXEC SQL FETCH c1 INTO...;  
if ( businessLogic is true)  
    EXEC SQL UPDATE employee SET job=:newjob WHERE CURRENT OF c1;  
EXEC SQL CLOSE c1;
```

# FOR UPDATE 文節 (Java)

- SELECT 文に FOR UPDATE 文節を指定する。
- getCursorName() を UPDATE 文の WHERE CURRENT OF に加える。

```
String sqlSelect = "SELECT C1, C2 FROM T1 FOR UPDATE";
String sqlUpdate = "UPDATE T1 SET C2=? WHERE CURRENT OF";
String cursorName = null;

Statement stmt = con.createStatement(sqlSelect);
ResultSet rs = stmt.executeQuery();
cursorName = rs.getCursorName();
PreparedStatement ps = con.prepareStatement(sqlUpdate + cursorName);

while (rs.next()) {
    String c1 = rs.getString(1);
    String c2 = rs.getString(2);

    if ( businessLogic is true ) {
        String newC2 = "new value";
        ps.setString(1, newC2);
        ps.executeUpdate();
    }
}
rs.close();
ps.close();
stmt.close();
```



# FETCH FIRST n ROWS ONLY 文節

## ■ パフォーマンス向上のために

- アプリケーションが n 行以上取り出すことを望まない場合に指定
  - 取り出す行が制限できるため、パフォーマンスの向上が見込まれます。
  - DB2 オプティマイザーが最初の n 行取り出す場合にふさわしいアクセスパスを選択します。
  - クライアント/サーバー間の転送データ・ブロックサイズに影響を与えます。

```
SELECT name, job, salary FROM staff  
WHERE deptno = 'D11'  
FETCH FIRST 10 ROWS ONLY
```

## ➤ 考慮点

- DB2 固有の書き方となります。
- FOR UPDATE と一緒に使用することはできません。

## 3.3. 開発を助けるツール

---

- 開発を助けるツール
  - コマンド行プロセッサ
  - コマンドウィンドウ
  - コマンドセンター

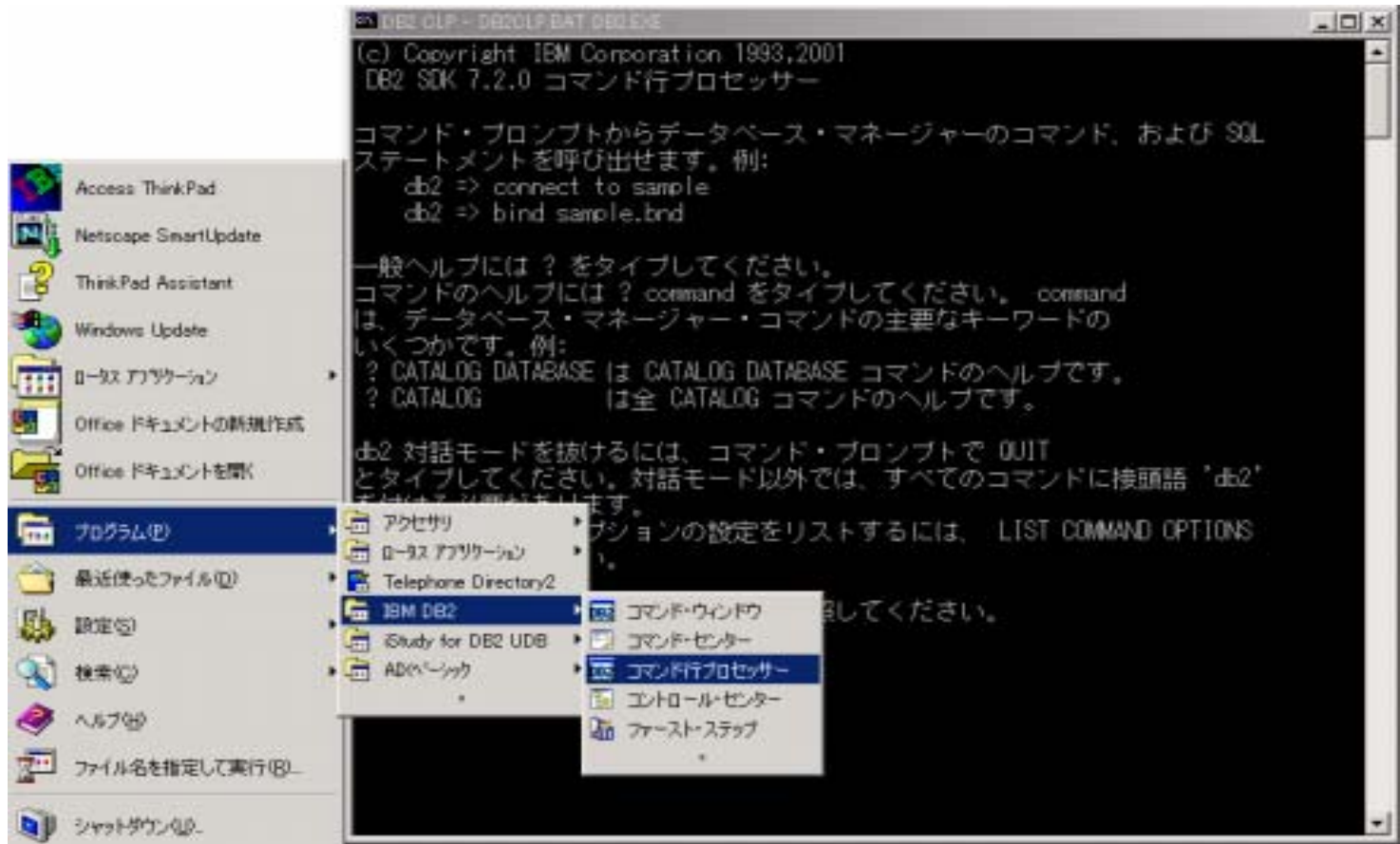
# コマンド行プロセッサ (CLP)

- コマンド行プロセッサとは
  - CLP (Command Line Processor) と呼ばれます。
  - DBMS に対してSQL ステートメントや DB2 コマンドをインタラクティブに実行するインタプリタです。
- 特徴
  - 以下の実行が可能です
    - SQL ステートメント
    - データベース・ユーティリティ
    - オンライン・ヘルプ
    - \*OS コマンドを実行する場合には先頭に感嘆符(!)をつけます
  - シェルコマンド(!)
    - UNIX ベースのシステム、OS/2、または Windows オペレーティング・システム上で、OSコマンドを対話式またはバッチ・モードで実行できるようになります
  - コマンド・オプション
    - 例

オプション	説明	デフォルト設定
-a	SQLCA を表示する	OFF
-c	自動コミット	ON
-f	コマンド入力をファイルから読取る	OFF

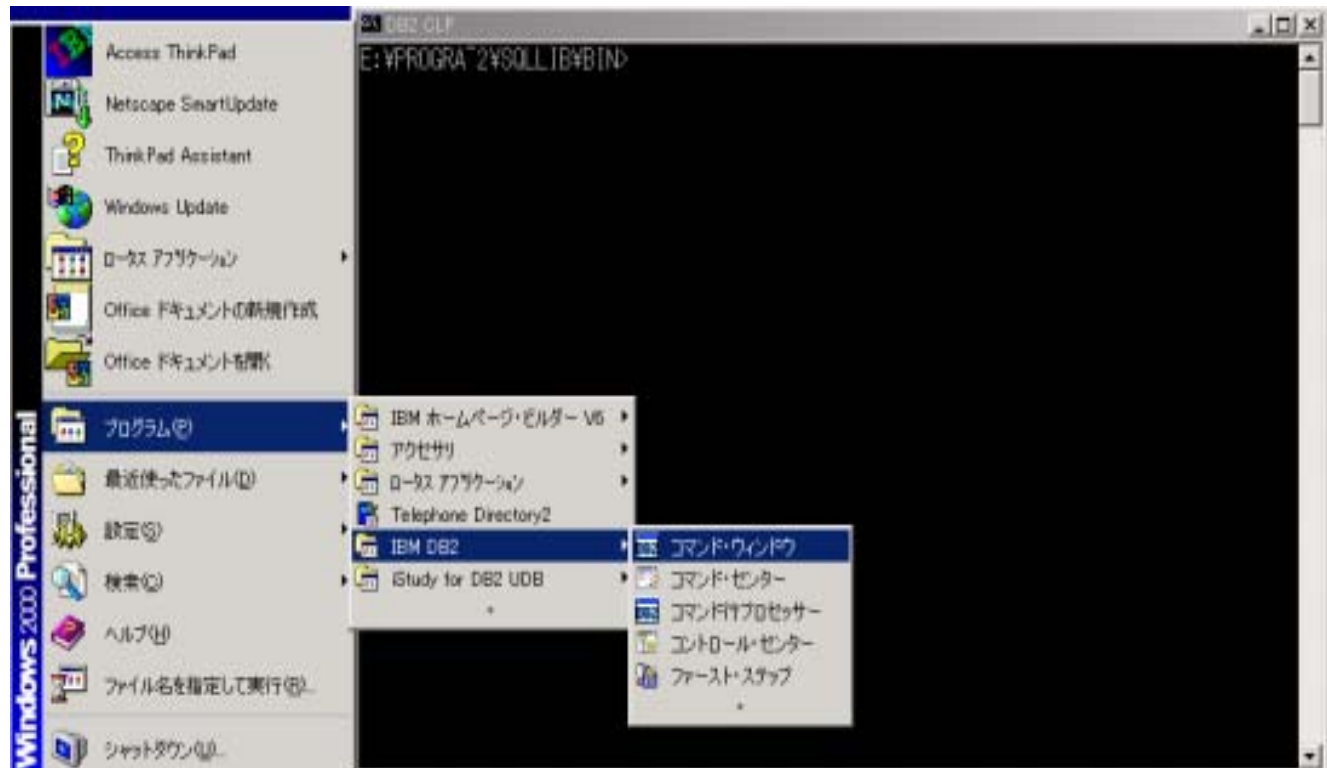
# コマンド行プロセッサ (CLP)

## ■ Windows の場合



# コマンドウィンドウ

- DB2 コマンドを入力する場合には、各コマンドの前に必ず "db2" と入力する。
- OS コマンドはそのまま実行できる。



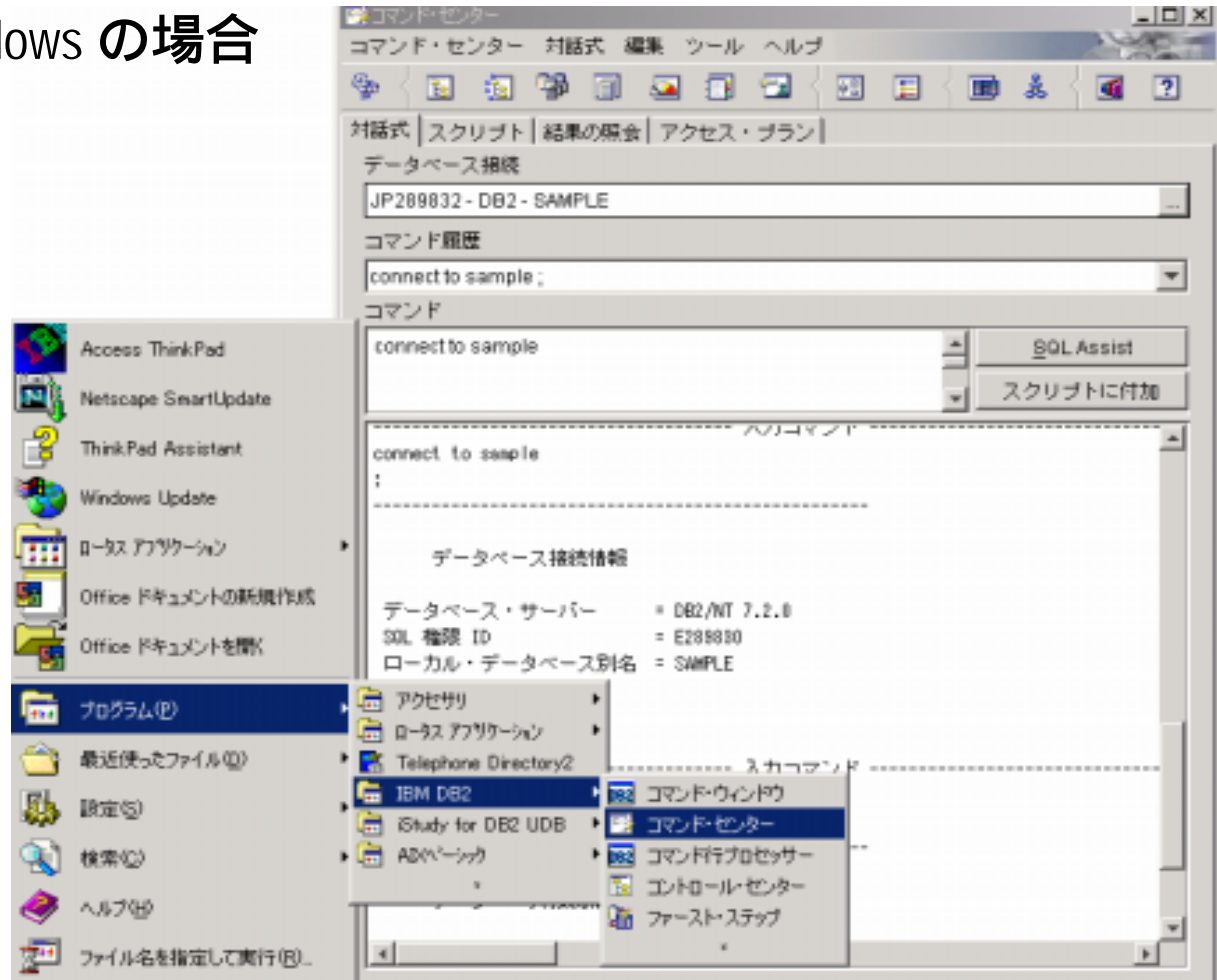
# コマンドセンター

---

- 対話型の GUI ツールです
- 特徴
  - 以下の実行が可能です
    - DB2 コマンド、SQL ステートメント、オンラインヘルプの実行
    - \*OSコマンドを実行する場合には、コマンドの先頭に感嘆符(!)をつけます
    - SQL Assist を使用して SQL ステートメントをマウスを使用して作成
    - コマンド・スクリプトでの作業(コマンドおよびSQL 文をスクリプトとして保管)
    - アクセス・プランのグラフィック表現の表示

# コマンドセンター

- Windows の場合



## 3.4. 開発ヒント集

---

- DB2 エラー
  - オンラインヘルプの使用方法
- 参考文献
  - マニュアル一覧
  - DB2 サポートハンドブック
  - その他の参考情報

# DB2 エラー

## エラーメッセージは、"XXXnnnnn"で表示

- XXX にはメッセージ接頭語が入ります。
  - CLI : コール・レベル・インターフェースで生成されるメッセージ
  - SQL : 警告やエラー状態が検出されたときにデータベース・マネージャーで生成されるメッセージ
    - メッセージの種類 N:エラー・メッセージ W:情報メッセージ C:重大なシステム・エラー・メッセージ
  - DB2: コマンド行プロセッサで生成されるメッセージ
- nnnnn はメッセージ番号を表します。
- SQLCODE
  - SQLCODE はメッセージ・タイプ (通知N、警告C、重大W) に応じて、正または負の番号としてアプリケーションに渡される。
  - W が正の値を持つのに対して、N と C は負の値を持つ。
  - DB2 は SQLCODE をアプリケーションに戻すので、アプリケーションは SQLCODE に関連するメッセージを受け取ることができる。
- SQLSTATE
  - SQL ステートメントの結果としての状況に対する値。

# オンラインヘルプの使用方法

- コマンド・ヘルプ : db2 "? command"

- db2 "? connect" と入力すると、CONNECT コマンドに関するヘルプが表示されます。

- db2 "? connect"

```
CONNECT [USER username [{USING password [NEW new-password CONFIRM  
confirm-password] |CHANGE PASSWORD}]
```

```
CONNECT RESET
```

```
CONNECT TO database-alias [IN {SHARE MODE | EXCLUSIVE MODE [ON  
SINGLE NODE]] [USER username [{USING password [NEW new-password  
CONFIRM confirm-password] | CHANGE PASSWORD}]
```

- メッセージ・ヘルプ : db2 "? XXXnnnn"

- db2 "? sql0014" と入力すると、メッセージ SQL0014 に関するヘルプを表示します。

- db2 "? sql0014"

SQL0014N ソース・ファイル名が無効です。

説明: 事前コンパイラーの呼び出しに指定したソース・ファイル名に無効な文字が含まれているか、またはソースス・ファイル名へのポインターが無効です。パッケージは作成されませんでした。

ユーザーの処置: 正しいソース・ファイル名を使用してください。

# マニュアル一覧 (1)

---

## ■ 管理の手引き

- データベースの設計、使用、および管理についての情報
- パフォーマンスを向上させるための、データベース環境の構成およびチューニングについての情報
  - <http://www.ibm.com/jp/software/data/developer/library/manual/db2online/db2d0/index.htm>

## ■ SQL解説書

- SQLステートメントに共通する SQL および言語要素の基本的な構文
- SQL ステートメント / SQL プロシージャーステートメント / 関数の構文図、意味の説明、規則、および例
  - <http://www.ibm.com/jp/software/data/developer/library/manual/db2online/db2s0/index.htm>

# マニュアル一覧 (2)

---

- コマンド解説書

- DB2 コマンドを使用する際に必要となる参照情報
  - <http://www.ibm.com/jp/software/data/developer/library/manual/db2online/db2m0/index.htm>

- メッセージ解説書

- DB2 の各種構成要素から戻されるエラー・メッセージをすべてリストしています。
  - <http://www.ibm.com/jp/software/data/developer/library/manual/db2online/db2m0/index.htm>

# DB2 サポートハンドブック

---

- 障害発生時に
  - 正確かつ短時間での障害の復旧
  - 障害の原因特定を早期にはかるために必要な情報の収集を容易に行うためのガイドです。
- この中で、よくある DB2 エラーを紹介しています。
- AIX 版
  - <http://www.ibm.com/jp/software/data/developer/library/techdoc/pdf/db2hbaixv1.pdf>
- Solaris 版
  - <http://www.ibm.com/jp/software/data/developer/library/techdoc/pdf/db2hbsolarisv1.pdf>

# その他の参考情報 (日本語版 Web サイト)

- DB2 Developer Domain
  - 新着情報、技術白書、最新記事、テクニカル情報、スキル育成、情報効果に関する情報を掲載しています。
    - <http://www.ibm.com/jp/software/data/developer/>
      - 「技術白書・DB2ファミリー関連」
        - » 「アプリケーション開発、プログラミング」カテゴリ内の各種資料
        - » 「カンタン!DB2テクテク第一歩」内の各種資料
  
- DB2 University Library
  - DB2 に関する技術情報を掲載しています。
    - <http://www.ibm.com/jp/software/data/db2univ/library>
      - DB2 UDB アクセス・プログラミング(for Java and MacroPSM)
      - JavaによるDBとWebアプリケーション連携入門
      - DB2+WebSphereによるWebサービス構築入門
      - DB2 UDB Windowsプログラミング入門
      - DB2 UDB(PC&UNIX) V7 運用管理ガイド/デザイン・ガイド
  
- DB2 フォーラム
  - DB2 技術者の情報交換フォーラムです。
    - <http://wsp01.alpha-mail.ne.jp/FRM/DB2>

# 第3章 まとめ

---

- 分離レベルの指定
  - 分離レベルの設定は、並行性とアプリケーションのパフォーマンスに影響を与えますが、両者はトレードオフの関係です。アプリケーション要件に応じて適切なレベルを設定するようにして下さい。
- より良い SQL 文の書き方
  - SQL の書き方はパフォーマンスに直接影響を与えるので、注意してください。
- 開発を助けるツール
- 開発ヒント集
  - 提供されているツール・情報を有効に活用しましょう。