

## 管理・モニタリング

<第2. 00版>2006年1月

お断り:当資料は、DB2 SQL Replication または DpropR V8 をベースに作成されています。

©日本IBM システムズ・エンジニアリング(株) インフォメーション・マネージメント



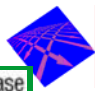
## 管理・モニタリング

*blank page*



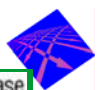
## 内容

- 管理・モニタリングでよく使用する制御表
- 日常監視する項目
- モニタリングFAQ
  - キャプチャーは何トランザクション処理したか
  - キャプチャーは何件CD表に書き出したか
  - キャプチャーのプルーニング状況はどうか
  - アプライがまだ適用していないのは何件か
  - アプライはどんな更新をターゲット表にしたか
  - アプライはどれくらい遅れているのか
- エラーは発生していないか
- 様々なトレース
  - TRCFLOW、TRCERR、TRCRWK、TRCPERF
  - ASNTRC
  - ASNANALYZE
- 整合性の確認方法
  - ASNTDIFF、ASNTREPユーティリティ
- レプリケーションセンターでのモニタリング
- レプリケーション・アラート・モニター



DB2 Universal Database

*blank page*



DB2 Universal Database

## キャプチャーの管理・モニタリングでよく使用される表

### ■ 管理・モニタリング項目例

- キャプチャーは何トランザクション処理したか
- キャプチャーは何件CD表に書き出したか
- キャプチャーのプルーニング状況はどうか

### ■ IBMSNAP\_CAPMON表

- キャプチャーのパフォーマンス操作統計
  - MONITOR\_INTERVALごとに更新
    - － デフォルト 5分
  - MONITOR\_LIMITに達するとプルーニングされる
    - － デフォルト 10080分(7日)

### ■ IBMSNAP\_CAPTRACE表

- キャプチャーの起動終了メッセージ
- キャプチャーからの通知、警告、およびエラーメッセージ
- PRUNE\_INTERVAL毎にプルーニングされる



DB2 Universal Database

## 解説: IBMSNAP\_CAPMON表

- MONITOR TIME
  - この表に行が挿入された時のキャプチャー・コントロール・サーバーのタイム・スタンプ
- RESTART TIME
  - キャプチャー・プログラムの現在の呼び出しが再始動されたときのタイム・スタンプ
- CURRENT MEMORY
  - キャプチャー・プログラムが使用したメモリーの量(バイト 単位)
- CD\_ROWS\_INSERTED
  - すべてのソース表について、キャプチャー・プログラムが CD 表に挿入した行数
- RECAP\_ROWS\_SKIPPED
  - Update-anywhere レプリケーションの場合、キャプチャー・プログラムが処理したが、CD 表に挿入していない行数
  - キャプチャー・プログラムの登録では、この表に複製された変更でも、このソース・サーバーから発生したものではない変更は、再キャプチャーしないよう定義されているため、これらの行はスキップされる
- TRIGR\_ROWS\_SKIPPED
  - キャプチャー・プログラムが処理したが、CD 表に挿入していない行数
  - キャプチャー・プログラムの登録で、特定の行を抑制するようにトリガーが定義されているため、これらの行はスキップされる
- CHG\_ROWS\_SKIPPED
  - キャプチャー・プログラムが処理したが、CD 表に挿入していない行数
  - キャプチャー・プログラムの登録では、登録済みの列で発生した変更のみをキャプチャーするように定義されているため、これらの行はスキップされる
- TRANS\_PROCESSED
  - キャプチャー・プログラムが処理した、ソース・システム上のトランザクションの数



DB2 Universal Database

## 解説:IBMSNAP\_CAPMON表 (つづき)

- TRANS\_SPILLED
  - メモリー制限のためにキャプチャー・プログラムがディスクに書き出した、ソース・システム上のトランザクションの数
- MAX\_TRAN\_SIZE
  - ソース・システムで発生した最大のトランザクション
  - トランザクション・サイズを知ることによって、メモリー・パラメーターの変更を検討することになる
- LOCKING\_RETRIES
  - デッドロックにより再処理が必要となった回数
- JRN\_LIB (OS/400)
  - キャプチャー・プログラムが処理していたジャーナルのライブラリー名
- JRN\_NAME (OS/400)
  - キャプチャー・プログラムが処理していたジャーナルの名前
- LOGREADLIMIT
  - 1000 のレコードが読み取られたが、それら 1000 のレコードの中に完了済みのトランザクションが見つからなかったために、キャプチャー・プログラムがログ・レコード読み取りを一時停止した回数。
- CAPTURE\_IDLE
  - 処理するものがないため、キャプチャー・プログラムがスリープした回数
- SYNCHTIME
  - この表にモニター・レコードが挿入されたときに、登録表のグローバル行から読み取られた SYNCHTIME の現行値



DB2 Universal Database

## 解説:IBMSNAP\_CAPTRACE表

- OPERATION
    - キャプチャー・プログラムの操作のタイプ
      - INFO、WARNING、ERROR
  - TRACE\_TIME
    - キャプチャー・トレース表に行が挿入されたときの、キャプチャー・コントロール・サーバーにおける時刻
  - DESCRIPTION
    - メッセージ ID とメッセージ・テキスト
    - エラー・メッセージ、警告メッセージ、または情報メッセージ
    - この列に入れられるテキストは英語のみ
- 以下はiSeriesの場合
- OPERATION
    - 初期化、キャプチャー、またはエラー条件など、キャプチャー・プログラムで実行された操作のタイプ
  - TRACE\_TIME
    - キャプチャー・トレース表に行が挿入された時刻
    - トレース限度整理の対象となる TRACE\_TIME 行は、キャプチャー・プログラムが CD 表および UOW 表の整理を行うときに削除される
  - JOB\_NAME
    - このトレース項目を書き込んだジョブの完全修飾名
    - 位置
      - 説明
    - 1-10
      - キャプチャー・スキーマ名またはジャーナル・ジョブ名
    - 11-20
      - キャプチャー・プログラムを始動したユーザーの ID
    - 21-26
      - ジョブ番号
  - JOB\_STR\_TIME
    - JOB\_NAME 列で指定されたジョブの開始時刻
  - DESCRIPTION
    - メッセージ ID とメッセージ・テキスト
    - メッセージ ID は DESCRIPTION 列の最初の 7 文字
    - メッセージ・テキストは、DESCRIPTION 列の位置 9 から始まる



DB2 Universal Database

## アプライのモニタリングでよく使用される表

### ■ IBMSNAP\_APPLYTRAIL表

- すべてのサブスクリプション・セット・サイクルの監査証跡情報
- サブスクリプションに対して実行された更新の履歴
- 問題判別診断およびパフォーマンス検証の際に有効
- プルーニングされない

### ■ IBMSNAP\_APPLYTRACE表

- アプライの起動終了メッセージ
- キャプチャーからの通知、警告、およびエラーメッセージ
- プルーニングされない



DB2 Universal Database

## IBMSNAP\_APPLYTRAIL表

### ■ APPLY\_QUAL

- アプライ修飾子

### ■ SET\_NAME

- アプライ・プログラムが処理したサブスクリプション・セットの名前

### ■ SET\_TYPE

- 最後のアプライ・サイクルの後でIBMSNAP\_SUBS\_SET表の SET\_TYPE 列に表示された値

### ■ WHOS\_ON\_FIRST

- Update-anywhereでは、処理順序をコントロールするために以下の値を使用
- F
  - (first の略) ソース表がレプリカであり、ターゲット表がマスター
- レプリカ表とマスター表の間で更新の矛盾が生じた場合、レプリカの側の矛盾するトランザクションはリジェクトされる
  - F は読み取り専用のサブスクリプションでは使用されない。Update-anywhere で使用されるもの
- S
  - (second の略) ソース表はマスター表またはその他のソースであり、ターゲット表はレプリカまたはその他のコピー
  - マスター表とレプリカ表の間で更新の矛盾が生じた場合、レプリカの側の矛盾するトランザクションはリジェクトされる
  - S は、すべての読み取り専用サブスクリプションについて使用される

### ■ ASNLOAD

- Y
  - パラメーター `loadxit=y` を指定してアプライ・プログラムを始動したため、サブスクリプション・セットのフル・リフレッシュを実行するために ASNLOAD ユーザー出力ルーチンが呼び出された
- N
  - フル・リフレッシュが必要ないか、アプライ・プログラムの始動時に `loadxit` パラメーターが指定されていないため、ASNLOAD 出力ルーチンが呼び出されない
- NULL
  - ASNLOAD 出力ルーチンを呼び出すかどうかをアプライ・プログラムが判断する前に、アプライ・プログラム・エラーが生じたことを示す



DB2 Universal Database

## IBMSNAP\_APPLYTRAIL表 (つづき)

- ★ FULL\_REFRESH
  - フル・リフレッシュが発生したかどうかを示すフラグ。
  - Y
    - サブスクリプション・セットに対してフル・リフレッシュが実行された
  - N
    - サブスクリプション・セットに対してフル・リフレッシュが実行されなかった
  - NULL
    - フル・リフレッシュが必要かどうかをアプライ・プログラムが判断する前に、エラーが生じた
- EFFECTIVE MEMBERS
  - 1 回のアプライ・サイクルで変更されたサブスクリプション・セット・メンバーの数
- ★ SET\_INSERTED
  - サブスクリプション・サイクルにおいてサブスクリプション・セット・メンバーに挿入された行数
- ★ SET\_DELETED
  - サブスクリプション・サイクルにおいてサブスクリプション・セット・メンバーから削除された行数
- ★ SET\_UPDATED
  - サブスクリプション・サイクルにおいてサブスクリプション・セット・メンバーで更新された行数
- ★ SET\_REWORKED
  - 最後のサイクルでアプライ・プログラムが再処理した行数。
  - 行がターゲット表にすでに存在しているためINSERTが失敗した場合、アプライ・プログラムは、INSERTをUPDATEに変換
  - 行がターゲット表に存在していないためUPDATEが失敗した場合、アプライ・プログラムは、UPDATEをINSERTに変換
- SET\_REJECTED\_TRXS
  - Update-anywhere 競合のためにリジェクトされたトランザクションの合計数
  - この列は、競合検出が「標準」または「詳細」と定義されている Update-anywhere サブスクリプション・セットに対してのみ使用される





## IBMSNAP\_APPLYTRAIL表 (つづき)

- ★ STATUS
  - 特定のサイクル後のアプライ・プログラムの作業状況を表す値。
  - -1
    - レプリケーションは失敗した
    - アプライ・プログラムは適用済みの行のセット全体をROLLBACKする
    - 始動パラメーターが SQLERRCONTINUE = Y の場合、最後のサイクル中にアプライ・プログラムに戻される SQLSTATE は、SQLERRCONTINUE ( *apply\_qualifier*.SQS) の入力ファイルでユーザーが指定した許容エラーの 1 つでは ない
  - 0
    - 正常に処理した
  - 2
    - サブスクリプション・セットを複数のサイクルで処理した
    - MAX\_SYNCH\_MINUTESに従って分割された 1 つの論理サブスクリプションを正常に処理した
  - 16
    - 正常に処理し 0 という状況に戻したが、ユーザーが SQLERRCONTINUE 始動パラメーターで ( *apply\_qualifier*.SQS で) 指定したいくつかの SQL エラーを検出したため、いくつかの行をリジェクトした
    - 失敗した行の詳細は、 *apply\_qualifier*.ERR ファイルで確認できる
  - 18
    - 複数のサイクルでサブスクリプション・セットを処理し、2 という状況に戻したが、ユーザーが SQLERRCONTINUE 始動パラメーターで ( *apply\_qualifier*.SQS で) 指定した SQL エラーのいくつかを検出したため、いくつかの行がリジェクトされた
    - 失敗した行の詳細は、 *apply\_qualifier*.ERR ファイルで確認できる





## IBMSNAP\_APPLYTRAIL表 (つづき)

-  **LASTRUN**
  - 最後のサブスクリプションが開始された概算の時刻
  - アプライ・プログラムは、サブスクリプション・セットを処理するたびに LASTRUN 値を設定する
  - アプライ・プログラムがサブスクリプション・セットの処理を開始する、アプライ・コントロール・サーバーにおけるおおよその時刻を示す
-  **LASTSUCCESS**
  - サブスクリプション・セットが最後に正常に処理されたときの、処理開始時点のアプライ・コントロール・サーバーの時間
- **SYNCHPOINT**
  - サブスクリプション・セットのデータの処理が、この同期点の値まで終了していることを示している
- **SYNCHTIME**
  - サブスクリプション・セットのデータの処理が、このタイム・スタンプまで終了していることを示している
- **SOURCE\_SERVER**
  - ソース表およびビューが定義されているデータベース名
- **SOURCE\_ALIAS**
  - SOURCE\_SERVER 列で指定されているデータベースの別名
- **SOURCE\_OWNER**
  - アプライ・プログラムが処理中であったソース表またはビューのスキーマ名
  - この値は、アプライ・サイクルが失敗したときにのみ設定される
- **SOURCE\_TABLE**
  - アプライ・プログラムが処理中であったソース表またはビューの名前
  - この値は、アプライ・サイクルが失敗したときにのみ設定される
- **SOURCE\_VIEW\_QUAL**
  - アプライ・プログラムが処理中であったソース表またはビューのソース・ビュー修飾子の値
  - この値は、アプライ・サイクルが失敗したときにのみ設定される



DB2 Universal Database

## IBMSNAP\_APPLYTRAIL表 (つづき)

- **TARGET\_SERVER**
  - ターゲットの表またはビューが保管されているサーバーのデータベース名
- **TARGET\_ALIAS**
  - TARGET\_SERVER 列で指定されているデータベース名の別名
- **TARGET\_OWNER**
  - アプライ・プログラムが処理中であったターゲット表のスキーマ名
  - この値は、アプライ・サイクルが失敗したときにのみ設定される
- **TARGET\_TABLE**
  - アプライ・プログラムが処理中であったターゲット表の名前
  - この値は、アプライ・サイクルが失敗したときにのみ設定される
- **CAPTURE\_SCHEMA**
  - このサブスクリプション・セットのキャプチャー・サーバー表のスキーマ名
- **TGT\_CAPTURE\_SCHEMA**
  - ターゲット表が別のサブスクリプション・セットのソースでもある場合 (multi-tier 構成中の外部の CCD 表、または Update-anywhere 構成中のレプリカ表など) は、この列には、表がソースとして機能するときに使用されるキャプチャー・スキーマが含まれる
- **FEDERATED\_SRC\_SRVR**
  - DB2 以外のリレーショナル・ソースの場合にのみアプライされる、サブスクリプション・セットのソースである、フェデレーテッド・リモート・サーバーの名前
- **FEDERATED\_TGT\_SRVR**
  - DB2 以外のリレーショナル・ターゲット・サーバーの場合にのみアプライされる、サブスクリプション・セットのターゲットである、フェデレーテッド・リモート・サーバーの名前



DB2 Universal Database

## IBMSNAP\_APPLYTRAIL表（つづき）

- JRN\_LIB
  - OS/400 キャプチャー・サーバーにのみアプライされる列
  - ソース表が使用するジャーナルのライブラリー名
- JRN\_NAME
  - OS/400 キャプチャー・サーバーにのみアプライされるこの列
  - ソース表が使用するジャーナルの名前
  - この列で、アスタリスクの後ろに 9 つの空白が続くときには、ソース表が現在ジャーナルの中にないことを意味する
  - この場合、このソース表のデータをキャプチャーすることはできない
- COMMIT\_COUNT
  - IBMSNAP\_SUBS\_SET表の中に記録される、最後のアプライ・サイクルからの COMMIT\_COUNT の値
- OPTION\_FLAGS
  - DB2 レプリケーションの将来のオプション用に予約済み
  - 現在、デフォルト値の NNNN が入っている
- EVENT\_NAME
  - セットの処理を起動したイベントを表すために使用されるユニークな文字ストリング
- ENDTIME
  - アプライ・プログラムがサブスクリプション・セットの処理を終了したときの、アプライ・コントロール・サーバーにおける時刻
  - セットの処理に要した時間を知るには、LASTRUN を ENDTIME から減算する
- SOURCE\_CONN\_TIME
  - アプライ・プログラムが初めてソース・データに接続したときの、キャプチャー・コントロール・サーバーにおける時刻



DB2 Universal Database

## IBMSNAP\_APPLYTRAIL表（つづき）

- SQLSTATE
  - 失敗した実行の SQLSTATEコード
  - それ以外の場合は、NULL
- SQLCODE
  - 失敗した実行の SQL エラー・コード
  - それ以外の場合は、NULL
- SQLERRP
  - 実行の失敗原因となった SQL エラーが生じたサーバーのデータベース製品 ID
  - それ以外の場合は、NULL
- ★ ■ SQLERRM
  - 失敗した実行の SQL エラーメッセージ
- APPERRM
  - アプライ・プログラムの実行に失敗したときのエラー・メッセージ ID およびメッセージ



DB2 Universal Database



## 解説:IBMSNAP\_APPLYTRACE

- APPLY\_QUAL
  - アプライ修飾子
- TRACE\_TIME
  - この表に行が挿入されたときのアプライ・コントロール・サーバーの時刻
- OPERATION
  - アプライ・プログラムの操作のタイプ
  - INFO、WARNING、ERROR
- DESCRIPTION
  - メッセージ ID とメッセージ・テキスト
  - メッセージ ID は DESCRIPTION 列の最初の 7 文字
  - メッセージ・テキストは、DESCRIPTION 列の位置 9 から始まる



*blank page*

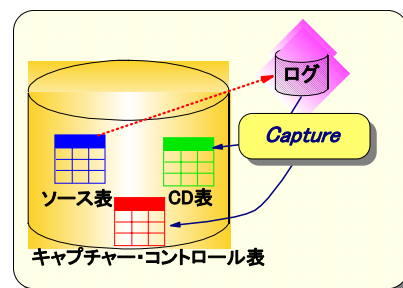


## 日常監視する項目

### ■ キャプチャープログラム

#### ● プロセス監視

- OS
  - ps コマンド
- DB2
  - list applications
  - キャプチャーのスレッド数分表示される
- レプリケーション・センター



#### ● キャプチャーの稼動確認

- REGISTER表のSYNCHTIMEを確認
  - DB2の更新がなくとも、REGISTER.SYNCHTIMEはCOMMIT\_INTERVAL毎に更新される
    - COMMIT\_INTERVALのデフォルトは30秒
  - CURRENT\_TIMESTAMPと比較することで キャプチャー がどれだけ遅れているかがわかる



DB2 Universal Database

## 解説:

IBMSNAP\_REGISTER

### ■ 実行例

```
$ ps -ef | grep -v grep | grep asn
yanav82 149194 111312 0 15:42:18 pts/11 0:00 asncap capture_server=TEST logstdout
```

実行コマンドも確認できる

```
$ db2 list applications
```

Auth Id	Application Name	Appl. Handle	Application Id	DB Name	# of Agents
YANAV82	asncap	667	*LOCAL.yanav82.050913064224	TEST	1
YANAV82	asncap	666	*LOCAL.yanav82.050913064223	TEST	1
YANAV82	asncap	665	*LOCAL.yanav82.050913064222	TEST	1
YANAV82	asncap	664	*LOCAL.yanav82.050913064221	TEST	1
YANAV82	asncap	662	*LOCAL.yanav82.050913064219	TEST	1

キャプチャーの5つのスレッドが表示される

- ・ HoldL Thread
- ・ Admin Thread
- ・ Worker Thread
- ・ Prune Thread
- ・ Monitor Thread

```
$ db2 "SELECT GLOBAL_RECORD, SYNCHPOINT, SYNCHTIME, CURRENT_TIMESTAMP as CURRENT_TIMESTAMP
FROM ASN. IBMSNAP_REGISTER WHERE GLOBAL_RECORD=' Y' WITH UR"
```

GLOBAL_RECORD	SYNCHPOINT	SYNCHTIME	CURRENT_TIMESTAMP
Y	x' 4360717C0000000030000'	2005-10-27-15. 19. 40. 000000	2005-10-27-15. 20. 27. 961000

1 レコードが選択されました。

COMMIT\_INTERVALごとにSYNCHTIMEが更新されていればキャプチャーは稼動している

GLOBAL_RECORD	SYNCHPOINT	SYNCHTIME	CURRENT_TIMESTAMP
Y	x' 4360717C0000000030000'	2005-10-27-15. 20. 35. 382000	2005-10-27-15. 20. 35. 712000

1 レコードが選択されました。



DB2 Universal Database

## 解説:

## IBMSNAP\_REGISTER

- SYNCTIME と CURRENT\_TIMESTAMP を比較することで キャプチャー がどれだけ遅れているかを確認できる

```
$db2 "SELECT GLOBAL_RECORD, SYNCHPOINT, SYNCHTIME, CURRENT_TIMESTAMP as CURRENT_TIMESTAMP,
(CURRENT_TIMESTAMP - SYNCHTIME) as LATE_TIME FROM ASN. IBMSNAP_REGISTER WHERE GLOBAL_RECORD='Y' WITH UR"
```

GLOBAL_RECORD	SYNCHPOINT	SYNCHTIME	CURRENT_TIMESTAMP	LATE_TIME
Y	x' 436072FC000000010000'	2005-10-27-15. 26. 04. 000000	2005-10-27-15. 26. 52. 544000	48. 544000

1 レコードが選択されました。

キャプチャーは約48秒遅れている

- キャプチャー のデフォルトのCOMMIT\_INTERVALは30秒のため、最大値は約30となる
- 30秒以上遅れていれば、キャプチャーが遅れている可能性がある



DB2 Universal Database

## 解説:

レプリケーション・センター

操作 - キャプチャー・コントロール・サーバー

名前	システム名	インスタンス	データベース名	タイプ	サーバー
QIT_M	9.188.198.48	MARRON	QIT	リモート	
SAMPLE	AHA02437	DB2	SAMPLE	ローカル	
TEST				ローカル	

状況のチェック

AHA02437 - DB2 - TEST

キャプチャー・スキーマ(C) ASN

リフレッシュ・インターバル(R) 5分

スレッド・タイプ	状況
HoldL スレッド	休止
Admin スレッド	休止
Prune スレッド	休止
Worker スレッド	休止

3項目中の3項目が表示されました



DB2 Universal Database

## 日常監視する項目

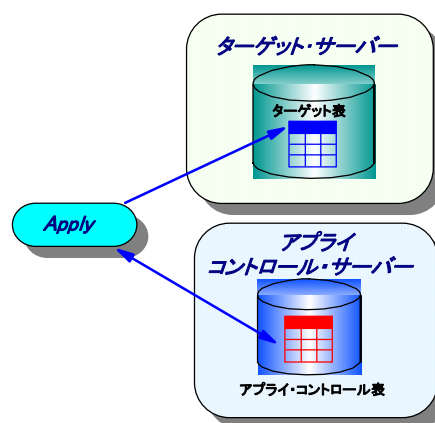
### ■ アプライプログラム

#### ● プロセス監視

- OS
  - ps コマンド
- DB2
  - list applications
  - アプライのスレッド数分表示される
- レプリケーション・センター

#### ● アプライ稼動確認

- APPLYTRAIL表
  - SLEEP\_MINUTES毎にアプライはAPPLYTRAIL表にINSERTを行う
  - STATUSのチェック
  - APPERRMの内容確認



## 解説:

### ■ 実行例

```
$ ps -ef | grep asnapp | grep -v grep
yanav82 156156 111312 5 17:17:35 pts/11 0:00 asnapply apply_qual=Q1 control_server=SAMPLE logstdout
```

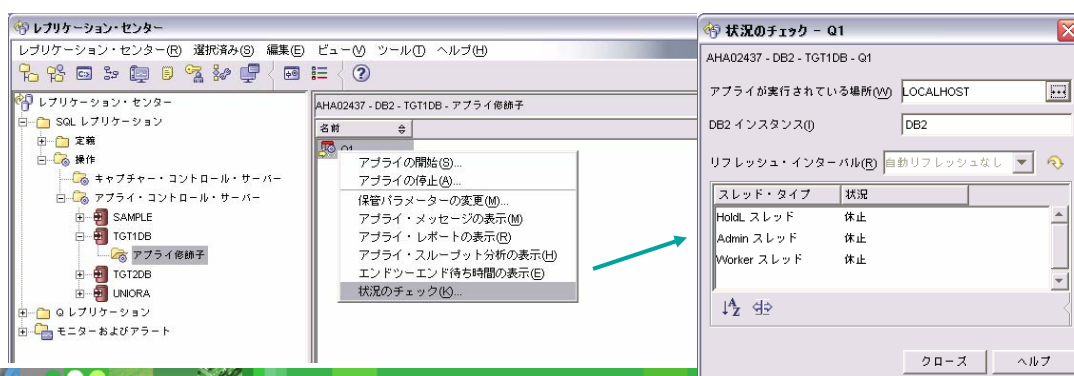
実行コマンドも確認できる

```
$ db2 list applications
```

Auth Id	Application Name	Appl. Handle	Application Id	DB Name	# of Agents
YANAV82	asnapply	798	*LOCAL. yanav82. 050913081738	SAMPLE	1
YANAV82	asnapply	797	*LOCAL. yanav82. 050913081737	SAMPLE	1
YANAV82	asnapply	795	*LOCAL. yanav82. 050913081735	SAMPLE	1

アプライの3つのスレッドが表示される

- ・ HoldL Thread
- ・ Admin Thread
- ・ Worker Thread



## 解説:

## IBMSNAP\_APPLYTRAIL

## ■ 実行例

```
$ db2 "SELECT APPLY_QUAL, SET_NAME, COUNT(*) as COUNT FROM ASN.IBMSNAP_APPLYTRAIL
WHERE LASTSUCCE >= '2005-09-13-17.00.00.00000' GROUP BY APPLY_QUAL, SET_NAME"
```

APPLY_QUAL	SET_NAME	COUNT
Q1	S1	6

1 record(s) selected.

```
$ db2 "SELECT APPLY_QUAL, SET_NAME, COUNT(*) as COUNT FROM ASN.IBMSNAP_APPLYTRAIL
WHERE LASTSUCCE >= '2005-09-13-17.00.00.00000' GROUP BY APPLY_QUAL, SET_NAME"
```

APPLY_QUAL	SET_NAME	COUNT
Q1	S1	7

1 record(s) selected.

```
$ db2 "SELECT APPLY_QUAL, SET_NAME, COUNT(*) as COUNT FROM ASN.IBMSNAP_APPLYTRAIL
WHERE LASTSUCCE >= '2005-09-13-17.00.00.00000' GROUP BY APPLY_QUAL, SET_NAME"
```

APPLY_QUAL	SET_NAME	COUNT
Q1	S1	8

1 record(s) selected.

SLEEP\_MINUTESごとに  
COUNTが更新されていれば  
アプライは稼動している



## 解説:

## IBMSNAP\_APPLYTRAIL

- STATUS=-1 ⇒ エラー
- APPERRM ⇒ アプライのエラーメッセージ

```
$ db2 "SELECT STATUS, substr(APPLY_QUAL, 1, 15) as APPLY_QUAL, substr(SET_NAME, 1, 5) as SET_NAME, LASTRUN,
substr(SOURCE_TABLE, 1, 10) as SOURCE_TABLE, substr(TARGET_TABLE, 1, 10) as TARGET_TABLE,
SQLCODE, APPERRM FROM ASN.IBMSNAP_APPLYTRAIL WHERE STATUS=-1 AND LASTRUN > '2005-08-05-17.50.00.000000' "
```

STATUS	APPLY_QUAL	SET_NAME	LASTRUN	SOURCE_TABLE	TARGET_TABLE	SQLCODE	APPERRM
-1	Q1	S1	2005-08-05-17.56.00.311002	SRC_MALL	TGSRC_MALL	-803	ASN1001E APPLY "Q1": "WorkerThr

ead。アプライ・プログラムは、SQL エラーを検出しました。ERRCODE は、"CF0107"。SQLSTATE/は "23505"。SQLCODE は "-803"。SQLERRM は "1-TAKAYA.TGSRC\_MALL"。SQLERRP は "SQLDMISR"。サーバー名は、""。表名は "TAKAYA.TGSRC\_MALL"。

STATUS = -1 :エラー  
STATUS >= 0 :通常

STATUS = -1 になった  
原因のSQLCODE

ASNxxxE : レプリケーションエラー

STATUS = -1 : エラーとなったサブスクリプション  
APPERRM : アプライのエラーメッセージ



## モニタリングFAQ : キャプチャー

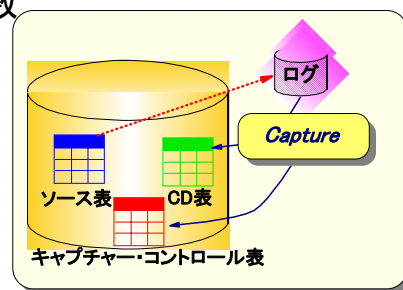
- キャプチャーは何トランザクション処理したの？
- キャプチャーは何件CD表にINSERTしたの？
- キャプチャーのプルーニング状況はどう？

### ■ IBMSNAP\_CAPMON表を参照

- キャプチャーが変更を収集しているかが分かる

#### 使用例

- ある時間からある時間まで変更を収集した件数
- 今日変更を収集した件数



DB2 Universal Database

## 解説:

IBMSNAP\_CAPMON

- ある時間からある時間までキャプチャーが変更を収集した件数

```
$ db2 "SELECT MONITOR_TIME, TRANS_PROCESSED, CD_ROWS_INSERTED FROM ASN. IBMSNAP_CAPMON
WHERE MONITOR_TIME > '2005-09-09-16.30.00.000000' AND MONITOR_TIME < '2005-09-09-17.30.00.000000'"
```

MONITOR_TIME	TRANS_PROCESSED	CD_ROWS_INSERTED
2005-09-09-16.32.19.958000	0	0
2005-09-09-16.37.20.570000	0	0
2005-09-09-16.42.20.942000	0	0
2005-09-09-16.47.21.254000	0	0
2005-09-09-16.52.21.676000	0	0
2005-09-09-16.57.21.928000	0	0
2005-09-09-17.02.22.330000	0	0
2005-09-09-17.07.22.822000	0	0
2005-09-09-17.12.23.264000	1	6

9 レコードが選択されました。

何トランザクションで  
何件処理したかがわかる

- TRANS\_PROCESSED
  - 処理したトランザクション数
- CD\_ROW\_INSERTED
  - CD表にINSERTされた行数
- ソース表を更新した分、CD\_ROWS\_INSERTEDへ件数が挿入されると、キャプチャーにより変更が収集されてことが分かる。
- ただし、MONITOR\_INTERVALに達しないとCAPMONへINSERTされない。
- この例では、MONITOR\_INTERVAL=5(5分)としているため、5分間にキャプチャーが処理したトランザクション数は1トランザクション、CD表へINSERTした件数は6件ということが分かる。

DB2 Universal Database



## 解説:

## IBMSNAP\_CAPMON

## ■ 今日キャプチャーが変更を収集した件数

```
$ db2 "SELECT DATE(MONITOR_TIME) AS YYYYMMDD, SUBSTR(CHAR(TIME(MONITOR_TIME)),1,2) AS HH,
SUM(TRANS_PROCESSED) AS TRANS_PROCESSED, SUM(CD_ROWS_INSERTED) AS CD_ROWS_INSERTED
FROM ASN. IBMSNAP_CAPMON WHERE MONITOR_TIME > '2005-09-09-00.00.00.000000' AND MONITOR_TIME < CURRENT_TIMESTAMP
GROUP BY DATE(MONITOR_TIME), SUBSTR(CHAR(TIME(MONITOR_TIME)),1,2)"
```

YYYYMMDD	HH	TRANS_PROCESSED	CD_ROWS_INSERTED
2005-09-09	14	0	0
2005-09-09	15	4	0
2005-09-09	16	2	0
2005-09-09	17	5	11
2005-09-09	18	0	0

時間ごとに合計何トランザクションで  
何件処理したかがわかる

5 レコードが選択されました。

- この例では、9月9日17時字の一時間に5トランザクションで11行変更がキャプチャーにより収集されたことがわかる。
- プルーニング処理もTRANS\_PROCESSEDにカウントされる
  - プルーニングではCAPMON、SIGNAL、CAPTRACE、CD表、UOW表などがDELETEされる
  - TRANS\_PROCESSED > 0 で CD\_ROW\_INSERTED = 0 ではプルーニングが行われたことがわかる



## 解説:

## IBMSNAP\_CAPTRACE

## ■ キャプチャーのプルーニング状況は？

- CAPTRACE表を使用
  - キャプチャーはIBMSNAP\_PRUNE\_SET.SYNCHTIME以前のデータをプルーニング対象とする
  - IBMSNAP\_PUNE\_SET.SYNCHTIMEはアプライによるターゲット表の更新がその時刻まで終了していることを表す

```
$ db2 "SELECT SUBSTR(DESCRIPTION,1,150) DESCRIPTION FROM ASN. IBMSNAP_CAPTRACE
WHERE OPERATION='INFO' AND TRACE_TIME > '2005-10-13-00.00.00.000000' | grep 'PruneThread'"
ASN01111 CAPTURE "ASN": "PruneThread". 枝取り循環は、"木 10 13 16:45:35 2005" に開始されました。

ASN01051 CAPTURE "ASN": "PruneThread". "8" 行は、表 "ASN"."IBMSNAP_CAPMON" から "木 10 13 16:45:35 2005" に枝取りされました。

ASN01051 CAPTURE "ASN": "PruneThread". "5" 行は、表 "ASN"."IBMSNAP_SIGNAL" から "木 10 13 16:45:35 2005" に枝取りされました。

ASN01051 CAPTURE "ASN": "PruneThread". "249" 行は、表 "ASN"."IBMSNAP_CAPTRACE" から "木 10 13 16:45:35 2005" に枝取りされました。

ASN01121 CAPTURE "ASN": "PruneThread". 枝取り循環が、"木 10 13 16:45:35 2005" に終了しました。

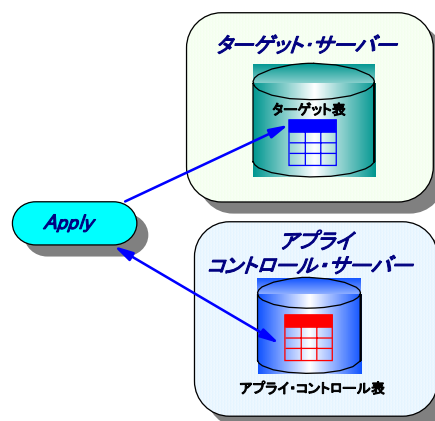
ASN01111 CAPTURE "ASN": "PruneThread". 枝取り循環は、"木 10 13 16:52:23 2005" に開始されました。
```

各表からいつ、何行DELETEされたかが確認できる



## モニタリングFAQ : アプライ

- CD表に溜まってるところどこまでアプライされたの？
- アプライがまだ適用していない最初のレコード(データ)は？
- アプライが行ったサブスクリプションセットごとの更新件数は？
  - ある時間からある時間まで
  - 今日起動してから今まで
- アプライが変更を適用しているかが分かる
  - CD表
  - IBMSNAP\_PRUNE\_SET表
  - IBMSNAP\_APPLYTRAIL表
- アプライはどれくらい遅れているのか
  - IBMSNAP\_REGISTER表
  - IBMSNAP\_PRUNE\_SET表



## 解説:

IBMSNAP\_PRUNE\_SET, CD

- CD表に溜まってるところどこまでアプライされたの？

```
$db2 "SELECT COUNT(*) FROM ASN. IBMSNAP_PRUNE_SET P, CD_SQLSRCTBL C
      WHERE C. IBMSNAP_COMMITSEQ > P. SYNCHPOINT AND P. SET_NAME='S1' AND P. APPLY_QUAL='Q1' "
```

```
1
-----
5
```

1 レコードが選択されました。

最もシンプル  
あるセットにだけ注目  
CD表にあるうち5件はまだアプライされていない

```
$db2 "(SELECT P. APPLY_QUAL, P. SET_NAME, COUNT(*) as COUNT FROM ASN. IBMSNAP_PRUNE_SET P, CD_SQLSRCTBL C
      WHERE C. IBMSNAP_COMMITSEQ > P. SYNCHPOINT AND P. SET_NAME='S1' GROUP BY P. APPLY_QUAL, P. SET_NAME)
      UNION
      (SELECT P. APPLY_QUAL, P. SET_NAME, COUNT(*) as COUNT FROM ASN. IBMSNAP_PRUNE_SET P, CDPRODUCT C
      WHERE C. IBMSNAP_COMMITSEQ > P. SYNCHPOINT AND P. SET_NAME='CCD' GROUP BY P. APPLY_QUAL, P. SET_NAME)"
```

APPLY_QUAL	SET_NAME	COUNT
Q1	S1	5
Q1	CCD	2

2 レコードが選択されました。

セットごとにCD表にあるうち  
まだアプライされていない件数が分かる

CD表. IBMSNAP\_COMMITSEQ : CD表にそのデータがコミットされた時  
IBMSNAP\_PRUNE\_SET. SYNCHPOINT : アプライによる最近のレプリケーション完了時点



## 解説:

## IBMSNAP\_PRUNE\_SET, CD

## ■ アプライがまだ適用していない最初のレコード(データ)は？

```
$db2 "SELECT P.APPLY_QUAL, P.SET_NAME, C.IBMSNAP_OPERATION, C.COL1 FROM ASN.IBMSNAP_PRUNE_SET P, CD_SQLSRCTBL C
WHERE C.IBMSNAP_COMMITSEQ > P.SYNCHPOINT AND P.SET_NAME='S1' ORDER BY C.IBMSNAP_COMMITSEQ ASC FETCH FIRST ROW ONLY"
```

APPLY_QUAL	SET_NAME	IBMSNAP_OPERATION	COL1
Q1	S1	I	20

1 レコードが選択されました。

```
$db2 "SELECT P.APPLY_QUAL, P.SET_NAME, C.IBMSNAP_OPERATION, C.COL1 FROM ASN.IBMSNAP_PRUNE_SET P, CD_SQLSRCTBL C
WHERE C.IBMSNAP_COMMITSEQ > P.SYNCHPOINT AND P.SET_NAME='S1' "
```

APPLY_QUAL	SET_NAME	IBMSNAP_OPERATION	COL1
Q1	S1	I	20
Q1	S1	I	30
Q1	S1	I	40
Q1	S1	I	50
Q1	S1	U	40

5 レコードが選択されました。

すべてを表示  
COL1はレプリケーションキー列

## ● IBMSNAP\_OPERATION

- I : INSERT
- U : UPDATE
- D : DELETE

## ● レプリケーションキー列も表示させることでどの列が更新されたかが分かる



## 解説:

## IBMSNAP\_APPLYTRAIL

## ■ アプライが適用したサブスクリプションセットごとの更新件数は？

## ● ある時間からある時間まで

```
$db2 "SELECT APPLY_QUAL, SET_NAME, SET_INSERTED, SET_UPDATED, SET_DELETED, SET_REWORKED FROM ASN.IBMSNAP_APPLYTRAIL
WHERE LASTRUN > '2005-09-09-17.50.00.000000' AND LASTRUN < '2005-09-09-18.00.00.000000' "
```

APPLY_QUAL	SET_NAME	SET_INSERTED	SET_UPDATED	SET_DELETED	SET_REWORKED
Q1	S1	4	1	0	0
Q1	CCD	0	0	0	0

2 レコードが選択されました。

## ● 今日起動してから今まで

```
$db2 "SELECT APPLY_QUAL, SET_NAME, sum(SET_INSERTED) as SET_INSERTED, sum(SET_UPDATED) as SET_UPDATED,
sum(SET_DELETED) as SET_DELETED, sum(SET_REWORKED) as SET_REWORKED FROM ASN.IBMSNAP_APPLYTRAIL
WHERE LASTRUN > '2005-09-09-00.00.00.000000' AND LASTRUN < CURRENT_TIMESTAMP GROUP BY APPLY_QUAL, SET_NAME"
```

APPLY_QUAL	SET_NAME	SET_INSERTED	SET_UPDATED	SET_DELETED	SET_REWORKED
Q1	CCD	22	1	0	0
Q1	S1	47	2	6	0

2 レコードが選択されました。

- SET\_INSERTED : ターゲット表にINSERTした件数
- SET\_UPDATED : ターゲット表にUPDATEした件数
- SET\_DELETED : ターゲット表にDELETEした件数
- SET\_REWORKED : REWORKが発生した件数



## 解説:

## IBMSNAP\_PRUNE\_SET, IBMSNAP\_REGISTER

## ■ アプライはどれくらい遅れているのか

```
$db2 "SELECT B.APPLY_QUAL, B.SET_NAME, A.SYNCHTIME AS CAPTURE_TIME, B.SYNCHTIME AS APPLY_TIME,
      (A.SYNCHTIME - B.SYNCHTIME) AS DIFF_TIME FROM ASN.IBMSNAP_REGISTER A, ASN.IBMSNAP_PRUNE_SET B
      WHERE A.GLOBAL_RECORD='Y' WITH UR"
```

APPLY_QUAL	SET_NAME	CAPTURE_TIME	APPLY_TIME	DIFF_TIME
Q1	S1	2005-10-13-17.12.25.256000	2005-10-13-17.11.53.200000	32.056000
Q1	CCD	2005-10-13-17.12.25.256000	2005-10-13-17.11.53.200000	32.056000

2 レコードが選択されました。

キャプチャーの  
COMMIT\_INTERVAL  
ごとに更新

アプライの  
SLEEP\_INTERVAL  
ごとに更新


SLEEP\_INTERVALによっては大きな  
数値となる

## ■ 差異が (COMMIT\_INTERVAL + SLEEP\_INTERVAL) より大幅に大きな数値となる場合、アプライが遅れている可能性が考えられる

```
$ db2 "SELECT B.APPLY_QUAL, B.SET_NAME, A.SYNCHTIME AS CAPTURE_TIME, B.SYNCHTIME AS APPLY_TIME,
      (A.SYNCHTIME - B.SYNCHTIME) AS APPLY_DIFF_TIME, (CURRENT_TIMESTAMP - A.SYNCHTIME) AS CAP_DIFF_TIME
      FROM ASN.IBMSNAP_REGISTER A, ASN.IBMSNAP_PRUNE_SET B WHERE A.GLOBAL_RECORD='Y' WITH UR"
```

APPLY_QUAL	SET_NAME	CAPTURE_TIME	APPLY_TIME	APPLY_DIFF_TIME	CAP_DIFF_TIME
Q1	S1	2005-10-13-17.21.30.610000	2005-10-13-17.12.57.332000	833.278000	50.823000
Q1	CCD	2005-10-13-17.21.30.610000	2005-10-13-17.13.29.388000	801.222000	50.823000

2 レコードが選択されました。



blank page

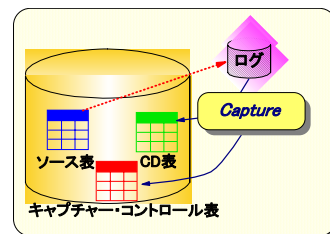
## エラーは発生していないか

### ■ キャプチャー

#### ● 実行ログ

- “インスタンス名.DB名.スキーマ名.CAP.log”
  - IBMSNAP\_CAPPARMS.CAPTURE\_PATH
- or
- asncap コマンド実行ディレクトリ

#### ● IBMSNAP\_CAPTRACE



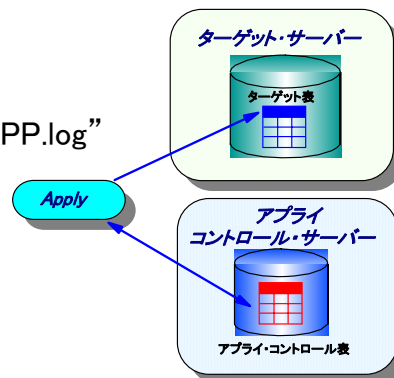
### ■ アプライ

#### ● 実行ログ

- “インスタンス名.DB名.アプライ修飾子名.APP.log”
  - IBMSNAP\_APPPARMS.APPLY\_PATH
- or
- asnapply コマンド実行ディレクトリ

- アプライ修飾子.TRC

#### ● IBMSNAP\_APPLYTRAIL



DB2 Universal Database

*blank page*

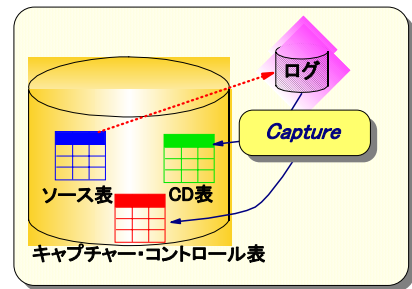
## エラーは発生していないか: キャプチャ

### ■ 実行ログ

- 起動パラメーター
- エラーログ
- プルーニング情報

### ■ IBMSNAP\_CAPTRACE表

- 実行ログと同等の内容が含まれる
- 10000件でラウンドロビンで再利用される



### 例

ログのデータが古すぎてキャプチャーがWarmスタートできなかった

```
$ db2 "SELECT * FROM ASN. IBMSNAP_CAPTRACE WHERE OPERATION='ERROR'"
```

OPERATION	TRACE_TIME	DESCRIPTION
ERROR	2005-09-13-15.41.15.099848 ASN0121E	CAPTURE "ASN" : "WorkerThread". The Capture program w arm start failed because existing data is too old. The Capture program will terminate.

1 record(s) selected.

DB2 Universal Database

## 解説:

### ■ 実行ログ

起動パラメーター

```
00 <setEnvDprRIB> ASN8003D "Capture": "" : "Initial": Program "capture 8.2.4" is starting.
00 <asnParmClass::printParms> ASN05291 "Capture": "ASN": "Initial": "CAPTURE_SERVER"の値は、起動時にメソッド "COMMANDLINI
00 <asnParmClass::printParms> ASN05291 "Capture": "ASN": "Initial": "CAPTURE_SCHEMA"の値は、起動時にメソッド "DEFAULT"に
00 <asnParmClass::printParms> ASN05291 "Capture": "ASN": "Initial": "LOGREUSE"の値は、起動時にメソッド "IBMSNAP CAPPARMS
00 <asnParmClass::printParms> ASN05291 "Capture": "ASN": "Initial": "LOGSDOUT"の値は、起動時にメソッド "COMMANDLINE"に
00 <asnParmClass::printParms> ASN05291 "Capture": "ASN": "Initial": "TERM"の値は、起動時にメソッド "IBMSNAP CAPPARMS"に
00 <asnParmClass::printParms> ASN05291 "Capture": "ASN": "Initial": "CAPTURE_PATH"の値は、起動時にメソッド "DEFAULT"に
00 <asnParmClass::printParms> ASN05291 "Capture": "ASN": "Initial": "DIAGLOG"の値は、起動時にメソッド "DEFAULT"に
00 <asnParmClass::printParms> ASN05291 "Capture": "ASN": "Initial": "ADD_PARTITION"の値は、起動時にメソッド "DEFAULT"に
00 <asnParmClass::printParms> ASN05291 "Capture": "ASN": "Initial": "SLEEP_INTERVAL"の値は、起動時にメソッド "IBMSNAP CAI
00 <asnParmClass::printParms> ASN05291 "Capture": "ASN": "Initial": "LAG_LIMIT"の値は、起動時にメソッド "IBMSNAP CAPPARM
00 <asnParmClass::printParms> ASN05291 "Capture": "ASN": "Initial": "COMMIT_INTERVAL"の値は、起動時にメソッド "IBMSNAP C
00 <asnParmClass::printParms> ASN05291 "Capture": "ASN": "Initial": "PRUNE_INTERVAL"の値は、起動時にメソッド "IBMSNAP CAI
00 <asnParmClass::printParms> ASN05291 "Capture": "ASN": "Initial": "RETENTION_LIMIT"の値は、起動時にメソッド "IBMSNAP C
00 <asnParmClass::printParms> ASN05291 "Capture": "ASN": "Initial": "AUTOPRUNE"の値は、起動時にメソッド "IBMSNAP CAPPARM
00 <asnParmClass::printParms> ASN05291 "Capture": "ASN": "Initial": "TRACE_LIMIT"の値は、起動時にメソッド "IBMSNAP CAPAI
00 <asnParmClass::printParms> ASN05291 "Capture": "ASN": "Initial": "MONITOR_LIMIT"の値は、起動時にメソッド "IBMSNAP CAPI
00 <asnParmClass::printParms> ASN05291 "Capture": "ASN": "Initial": "MONITOR_INTERVAL"の値は、起動時にメソッド "IBMSNAP
00 <asnParmClass::printParms> ASN05291 "Capture": "ASN": "Initial": "MEMORY_LIMIT"の値は、起動時にメソッド "IBMSNAP CAP
00 <asnParmClass::printParms> ASN05291 "Capture": "ASN": "Initial": "AUTOSTOP"の値は、起動時にメソッド "IBMSNAP CAPPARM
00 <asnParmClass::printParms> ASN05291 "Capture": "ASN": "Initial": "STARTMODE"の値は、起動時にメソッド "COMMANDLINE"に
00 <asnenv::setEnvDprCrcVhdl> ASN05941 "Capture": "ASN": "Initial" プログラムは、キーが (OSSEIPC48tempDB2.TEST.ASN.CAP.IPC
00 <WorkerMain> ASN1001 CAPTURE "ASN": "WorkerThread". キャプチャー・プログラムの初期化が成功しました。
00 <WorkerMain> ASN1009 CAPTURE "ASN": "WorkerThread". キャプチャー・プログラムは正常に初期化され、"0"登録に対するデータ
00 <handleCAPSTART> ASN0069W CAPTURE "ASN": "WorkerThread". IBMSNAP_PRUNCNL 表の SYNCHPOINT フィールドが、MAP ID "2
00 <handleCAPSTART> ASN0104I CAPTURE "ASN": "WorkerThread". MAP ID "2"を持つ CAPSTART シグナルへの応答で、ソースマ
00 <handleCAPSTART> ASN0069W CAPTURE "ASN": "WorkerThread". IBMSNAP_PRUNCNL 表の SYNCHPOINT フィールドが、MAP ID "3
00 <handleCAPSTART> ASN0104I CAPTURE "ASN": "WorkerThread". MAP ID "3"を持つ CAPSTART シグナルへの応答で、ソースマ
00 <handleCAPSTART> ASN0069W CAPTURE "ASN": "WorkerThread". MAP ID "4"を持つ CAPSTART シグナルへの応答で、ソースマ
00 <handleCAPSTART> ASN0104I CAPTURE "ASN": "WorkerThread". MAP ID "4"を持つ CAPSTART シグナルへの応答で、ソースマ
00 <handleCAPSTART> ASN0069W CAPTURE "ASN": "WorkerThread". MAP ID "5"を持つ CAPSTART シグナルへの応答で、ソースマ
00 <PruneMain> ASN1011 CAPTURE "ASN": "PruneThread". 枝取り循環は、"月 10 17 17:55:32 2005"に開始されました。
00 <PruneMain> ASN1011 CAPTURE "ASN": "PruneThread". 1行は、表 "ASN"."IBMSNAP_SIGNAL"から"月 10 17 17:55:32 2005"に
00 <PruneMain> ASN1011 CAPTURE "ASN": "PruneThread". 枝取り循環は、"月 10 17 17:55:32 2005"に終了しました。
00 <handleCAPSTART> ASN0069W CAPTURE "ASN": "WorkerThread". IBMSNAP_PRUNCNL 表の SYNCHPOINT フィールドが、MAP ID "4
00 <handleCAPSTART> ASN0104I CAPTURE "ASN": "WorkerThread". MAP ID "4"を持つ CAPSTART シグナルへの応答で、ソースマ
00 <handleCAPSTART> ASN0069W CAPTURE "ASN": "WorkerThread". IBMSNAP_PRUNCNL 表の SYNCHPOINT フィールドが、MAP ID "5
00 <handleCAPSTART> ASN0104I CAPTURE "ASN": "WorkerThread". MAP ID "5"を持つ CAPSTART シグナルへの応答で、ソースマ
00 <PruneMain> ASN1011 CAPTURE "ASN": "PruneThread". 枝取り循環は、"月 10 17 18:00:33 2005"に開始されました。
00 <PruneMain> ASN1011 CAPTURE "ASN": "PruneThread". 1行は、表 "ASN"."CDPRODUCT"から"月 10 17 18:00:33 2005"に
00 <PruneMain> ASN1011 CAPTURE "ASN": "PruneThread". 枝取り循環は、"月 10 17 18:00:33 2005"に終了しました。
00 <PruneMain> ASN1011 CAPTURE "ASN": "PruneThread". 枝取り循環は、"月 10 17 18:05:33 2005"に開始されました。
00 <PruneMain> ASN1011 CAPTURE "ASN": "PruneThread". 1行は、表 "TAKAYA"."CDPRODUCT"から"月 10 17 18:05:33 2005"に
```

プルーニング状況

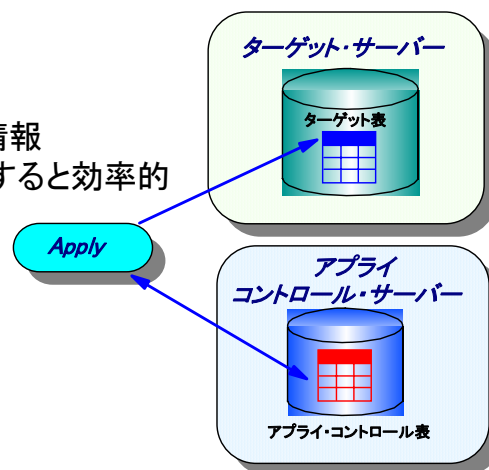
DB2 Universal Database



## エラーは発生していないか: アプライ

### ■ アプライ

- 実行ログ
  - 起動パラメーター
  - エラーログ
- IBMSNAP\_APPLYTRAIL表
  - エラー情報
    - ASNDONE exit プログラム
  - サブスクリプションセットの監査証跡情報
  - パフォーマンスや問題判別時に利用すると効率的
  - プルーニングされない



DB2 Universal Database

## 解説:

IBMSNAP\_APPLYTRAIL

### 例

トランザクション・ログがフルリになった場合と制約違反でターゲットへ変更を適用できなかった場合

```
$ db2 "SELECT STATUS, substr(APPLY_QUAL, 1, 15) as APPLY_QUAL, substr(SET_NAME, 1, 5) as SET_NAME, LASTRUN,
      substr(SOURCE_TABLE, 1, 10) as SOURCE_TABLE, substr(TARGET_TABLE, 1, 10) as TARGET_TABLE,
      SQLCODE, APPERRM FROM ASN. IBMSNAP_APPLYTRAIL WHERE STATUS=-1"
```

STATUS	APPLY_QUAL	SET_NAME	LASTRUN	SOURCE_TABLE	TARGET_TABLE	SQLCODE	APPERRM
-1	Q1	S2	2005-02-25-09.56.31.831001	SRC_MALL	TG_SMALL2	-964	ASN1001E APPLY "Q1": "WorkerThread". アプライ・プログラムは、SQL エラーを検出しました。ERRCODE は、"CF0107". SQLSTATE は "57011". SQLCODE は "-964". SQLERRM は "...". SQLERRP は "SQLR11SR". サーバー名は、"". 表名は "TAKAYA.TG_SMALL2".
-1	Q1	S1	2005-08-05-17.56.00.311002	SRC_MALL	TGSRC_MALL	-803	ASN1001E APPLY "Q1": "WorkerThread". アプライ・プログラムは、SQL エラーを検出しました。ERRCODE は、"CF0107". SQLSTATE は "23505". SQLCODE は "-803". SQLERRM は "1-TAKAYA.TGSRC_MALL". SQLERRP は "SQLDM1SR". サーバー名は、"". 表名は "TAKAYA.TGSRC_MALL".

STATUS = -1 : エラーとなったサブスクリプション  
 APPERRM : アプライのエラーメッセージ

DB2 Universal Database

## 解説:

ASNDONE exit

## ■ ASNDONE

- サブスクリプションセット実行後に毎回呼び出されるプログラム

## ■ アプライ起動時に以下のオプションを指定

- asnapply ... notify = y

## ■ ASNDONE使用例案

- STATUSが -1 になったときにだけ障害用のプログラムを実行させる

## ■ ASNDONEサンプルプログラム

- ~/sqlib/samples/repl/asndone.smp
- ユーザのPATHに登録されたディレクトリに保存
- スクリプトでも可能

## ■ ASNDONEで利用できる引数

- SET\_NAME
- APPLY\_QUAL
- WHOS\_ON\_FIRST
- CONTROL\_SERVER
- STATUS



## 解説:

IBMSNAP\_APPLYTRAIL

## ■ アプライ

- ASN.IBMSNAP\_APPLYTRAILの他の使用方法
- アプライがその日、各サブスクリプションセットを処理するのにかった最長時間

```
$ db2 "SELECT APPLY_QUAL, SET_NAME, MAX(ENDTIME - LASTRUN) AS TIME FROM ASN.IBMSNAP_APPLYTRAIL
WHERE LASTRUN > '2005-09-09-00.00.00.000000' AND LASTRUN < CURRENT_TIMESTAMP GROUP BY SET_NAME, APPLY_QUAL"
```

APPLY_QUAL	SET_NAME	TIME
Q1	CCD	3.935001
Q1	REP	0.380999
Q1	S1	7.801000
Q2	SET	0.981998

4 レコードが選択されました。

ENDTIME : サブスクリプションセット終了時間  
 LASTRUN : サブスクリプションセット開始時間

- アプライのその日の最大エンドツーエンド待ち時間

```
$ db2 "SELECT APPLY_QUAL, SET_NAME, MAX( (ENDTIME - LASTRUN) + (SOURCE_CONN_TIME - SYNCHTIME) ) AS TIME
FROM ASN.IBMSNAP_APPLYTRAIL WHERE LASTRUN > '2005-09-09-00.00.00.000000' AND LASTRUN < CURRENT_TIMESTAMP
GROUP BY SET_NAME, APPLY_QUAL"
```

APPLY_QUAL	SET_NAME	TIME
Q1	CCD	1946.836002
Q1	REP	9013324.820998
Q1	S1	7054403.086999
Q2	SET	0.960999

4 レコードが選択されました。

ENDTIME : サブスクリプションセット終了時間  
 LASTRUN : サブスクリプションセット開始時間  
 SOURCE\_CONN\_TIME : キャプチャーサーバー接続時間  
 SYNCHTIME : CD表へのコミット時間



## 内容

- 日常監視する項目
  - プロセス監視
  - 稼動確認
- モニタリングFAQ
  - キャプチャーは何件トランザクション処理したか
  - キャプチャーは何件CD表に書き出したか
  - キャプチャーのプルーニング状況はどうか
  - アプライがまだ適用していないのは何件か
  - アプライはどんな更新をターゲット表にしたか
  - アプライはどれくらい遅れているのか
- エラーは発生していないか
  - キャプチャー
  - アプライ
- 様々なトレース
  - TRCFLOW、TRCERR、TRCRWK、TRCPERF
  - ASNTRC
  - ASNANALYZE
- 整合性の確認方法
  - ASNTDIFF
  - ASNTREP
- レプリケーションセンターでのモニタリング
- レプリケーション・アラート・モニター



DB2 Universal Database

*blank page*



DB2 Universal Database

## 様々なトレース

### ■ アプライ起動パラメータ ( undocumented )

- TRCFLOW
  - アプライの実行トレースを画面に出力
- TRCERR
  - エラーのみ画面に出力
- TRCRWK
  - REWORK行の詳細情報を画面またはTRCファイルに出力
    - 今まではINSERT⇒UPDATEまたUPDATE⇒INSERTのどちらかの変換か判別できない場合があった
- TRCPERF
  - パフォーマンスの詳細情報を画面またはTRCファイルに出力
  - SET、MEMBERのレベルで情報を収集
  - アプライのパフォーマンスへは大きなインパクトはない

### ■ ASNTRC

### ■ ASNANALYZE



DB2 Universal Database

*blank page*



DB2 Universal Database

## 様々なトレース: TRCFLOW

### ■ TRCFLOW

- アプライ起動時に指定  
asnapply ... trcfLOW

トレース結果は標準出力に表示される  
テスト期間中に使用すると有効

```

:
PSET: CURRTSRVRTYPE is SQL, CURRTSRVRVERSION is 08
SAT: ASNLOAD = N, EFFECT_MEMBERS = 0
SAT: FULL_REFRESH = N
SAT: SET_INSERTED = 0
SAT: SET_DELETED = 0
SAT: SET_UPDATED = 0
SAT: SET_REWORKED = 0
SAT: SET_REJECTED_TRXS = 0
SAT: STATUS = 0
SAT: LASTRUN = 2005-10-04-16.22.44.289907
SAT: LASTSUCCESS = 2005-10-04-16.22.39.275764
SAT: SYNCHPOINT is 43422b4d000000010000
SAT: SYNCHTIME is 2005-10-04-16.22.14.145179
SAT: SOURCE_ALIAS is TEST
SAT: SOURCE_SERVER is TEST
SAT: SOURCE_OWNER is
SAT: SOURCE_TABLE is
SAT: TARGET_ALIAS is SAMPLE
SAT: TARGET_SERVER is SAMPLE
SAT: TARGET_OWNER is
SAT: TARGET_TABLE is
SAT: SQLSTATE is null
SAT: SQLERRM is null
SAT: SQLCODE is null
SAT: SQLERRP is null
SAT: APPERRM is null
ApplyWorker: mbrStates = 4
--- Process next subscription (2) ---
MAIN: CURRTSRVRTYPE is SQL, CURRTSRVRVERSION is 08
:

```

FULL\_REFRESH  
SET\_INSERTED  
SET\_DELETED  
SET\_UPDATED  
SET\_REWORKED  
STATUS

DB2 Universal Database



## 様々なトレース: TRCERR

### ■ TRCERR

- アプライ起動時に指定
  - デフォルトで暗黙的に指定されている
  - asnapply ... trcerr

”アプライ修飾子.TRC”ファイルに出力される  
TRCファイルは上書きされるため、ファイルのRenameをしよう！  
TRCFLOWと一緒に指定するとTRCファイルに出力されない

Apply program compiled at 13:25:32 on Aug 12 2005 (Level 80124m\_FP10)

SQLCA is

```

+00000000 53514c43 41202020 88000000 2dfdffff | SQLCA - |
+00000010 28005441 4b415941 2e545249 4731ff2d | ( TAKAYA. TRIG1 - |
+00000020 383033ff 32333530 35ff317c 54414b41 | 803 23505 1|TAKA |
+00000030 59412e54 52494754 424c2020 20202020 | YA. TRIGTBL |
+00000040 20202020 20202020 20202020 20202020 | |
+00000050 20202020 20202020 53514c44 4d495352 | SQLDMISR |
+00000060 6d001280 05000000 00000000 00000000 | m |
+00000070 00000000 00000000 20202020 20202020 | |
+00000080 20202030 39303030 | 09000 |

```

\*\*\* SQL ERROR \*\*\*: SQL0723N トリガー "TAKAYA. TRIG1" のトリガー SQL ステートメントでエラーが発生しました。 SQLCODE "-803"、SQLSTATE "23505" およびメッセージ・トークン "1|TAKAYA. TRIGTBL" を含むエラーの情報が戻りました。 SQLSTATE=09000

AINS2: insert\_string is INSERT INTO "TAKAYA"."TGPRODUCT" ("ID", "NAME", "PRICE", IBMSNAP\_OPERATION, IBMSNAP\_INTENTSEQ, IBMSNAP\_CO  
MMITSEQ, IBMSNAP\_LOGMARKER) VALUES ( ?, ?, ?, ?, ? )

AINS2: sqlda for INSERT statement

sqldaaid:

+00000000 00000000 002b0000

sqldabc: 1952

sqln: 44

sqld: 7

:

SQL ERROR : SQLCODE = -803

DB2 Universal Database



## 様々なトレース:TRCRWK

### ■ TRCRWK

- アプライ起動時に指定

asnapply ... trcrwk

“アプライ修飾子.TRC”ファイルに出力される

```
$ cat Q1.TRC
trace file name is /home/yanav82/DpropR/Q1.TRC
Apply program compiled at 06:43:52 on Aug 12 2005 (Level 80124m_FP10)
--- REWORK UPDATE INTO INSERT FOR SET(S1) MEMBER(0) SRC(YANAV82.SRCTBL.0) TRG(YANAV82.TGTBL) ---
sqldaid:
+00000000 00000000 00000000
sqldabc: 1952
sqln: 44
sqld: 2
SQLVAR[0].sqltype: 496
SQLVAR[0].sqlind = none (even SQLTYPE)
SQLVAR[0].sqllen: 4,
sqlvar[0].sqldata:
+00000000 00000064
SQLVAR[0].sqlname.length: 4
sqlvar[0].sqlname.data:
+00000000 434f4c31
SQLVAR[1].sqltype: 453
SQLVAR[1].sqlind: 0,
SQLVAR[1].sqllen: 20,
sqlvar[1].sqldata:
+00000000 41474149 4e202020 20202020 20202020
+00000010 20202020
SQLVAR[1].sqlname.length: 4
sqlvar[1].sqlname.data:
+00000000 434f4c32
```

UPDATEがINSERTにREWORKされてた



DB2 Universal Database

*blank page*



DB2 Universal Database



## 様々なトレース:TRCPERF

### ■ TRCPERF

- アプライ起動時に指定  
asnapply ... trcperf
- フォーマット

Set level:

- 1) record type (S)
  - 2) set name
  - 3) cycle counter
  - 4) who's on first (S or F)
  - 5) time before cntl server connect
  - 6) time after cntl server connect
  - 7) time before source server connect
  - 8) time after source server connect
  - 9) time before target server connect
  - 10) time after target server connect
  - 11) time before opening spill file \*
  - 12) time after closing spill file \*
  - 13) time before the final source server connect
  - 14) time after the final source server connect
- \* (for replica or transactional apply only)

Member level:

- 1) record type (M)
- 2) set name
- 3) cycle counter
- 4) member number
- 5) time prior to prepare
- 6) time prior to open cursor
- 7) time prior to the first fetch
- 8) time prior to the last fetch
- 9) time before opening spill file
- 10) time before closing spill file
- 11) # of rows inserted
- 12) # of rows updated
- 13) # of rows deleted
- 14) # of rows reworked



DB2 Universal Database

## 解説:

### ■ アプライ修飾子.TRC ファイルに結果が出力される

```
$ cat Q1.TRC
trace file name is F:\test\SmartAnswer\DpropR\Q1.TRC
Apply program compiled at 13:25:32 on Aug 12 2005 (Level 80124m_FP10)
S, S1, 1, S, 2005/10/31 11:43:49, 2005/10/31 11:43:49, 2005/10/31 11:43:51, 2005/10/31 11:43:52, 2005/10/31 11:43:53,
2005/10/31 11:43:55, , , 2005/10/31 11:43:57, 2005/10/31 11:43:59
M, S1, 1, 0, 2005/10/31 11:43:53, 2005/10/31 11:43:53, 2005/10/31 11:43:53, 2005/10/31 11:43:53, 2005/10/31 11:43:56,
2005/10/31 11:43:57, 0, 0, 51, 0
S, CCD, 2, S, 2005/10/31 11:44:03, 2005/10/31 11:44:03, 2005/10/31 11:44:03, 2005/10/31 11:44:05, 2005/10/31 11:44:05,
2005/10/31 11:44:08, , , 2005/10/31 11:44:10, 2005/10/31 11:44:11
M, CCD, 2, 1, 2005/10/31 11:44:05, 2005/10/31 11:44:05, 2005/10/31 11:44:05, 2005/10/31 11:44:05, 2005/10/31 11:44:09,
2005/10/31 11:44:10, 20, 0, 0, 0
```

④ - ③ = メンバー処理時間

③: prepare開始時間

④: メンバーのspill fileをcloseした時間

② - ① = サブスクリプションセット処理時間

①: ソース表への接続時間

②: 最後のメンバーのspill fileをcloseした時間



DB2 Universal Database

## 様々なトレース:ASNTRC

### ■ASNTRCユーティリティ

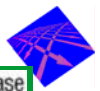
- 以下のプログラム・フローをログに記録
  - SQLキャプチャー・プログラム
  - SQLアプライ・プログラム
  - アラート・モニタープログラム
- 問題分析時に使用
  - 実行ログやモニター表では解析できないような障害時などに使用
  - 出力をIBMソフトウェア・サポートに提出する

### ■Linux、Unix、Windows、z/OS(USS環境)で実行

### ■ASNTRCコマンドを実行する

- 開始する (asntrc on)
- 停止する (asntrc off)
- トレース結果を表示する
  - プログラム・フローの表示 (asntrc flw)
  - すべてのデータ構造の表示 (asntrc fmt)

### ■キャプチャー、アプライ・プログラムの再起動は不要

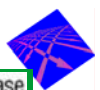


DB2 Universal Database

## 解説:

### ■ASNTRC使用例

- スキーマASNのキャプチャーのトレースを開始し結果をasncap.trcファイルに出力する
  - asntrc on -db MYDB -cap -schema ASN -fn asncap.trc
- スキーマASNのキャプチャーのトレースを停止する
  - asntrc off -db MYDB -cap -schema ASN
- 生成されたバイナリファイルasncap.trcからフォーマットする
  - asntrc flw -fn asncap.trc
- 生成されたバイナリファイルasncap.trcからパフォーマンス・データをフォーマットする
  - asntrc fmt -fn asncap.trc -d 4
- 指定可能なオプションを表示する
  - asntrc -help



DB2 Universal Database

## 解説:

- on
  - 特定の Q キャプチャー・プログラム、Q アプライ・プログラム、キャプチャー・プログラム、アプライ・プログラム、またはレプリケーション・アラート・モニター・プログラムについて、トレース機能をオンにすることを指定します。トレース機能は、トレース処理中に使用する共有メモリー・セグメントを作成します。
- -db b\_name
  - トレースするデータベースの名前を指定します。
  - トレースする Q キャプチャー・プログラム用の Q キャプチャー・サーバーの名前を指定します。トレースする Q アプライ・プログラム用の Q アプライ・サーバーの名前を指定します。| トレースするキャプチャー・プログラム用のキャプチャー・コントロール・サーバーの名前を指定します。| トレースするアプライ・プログラム用のアプライ・コントロール・サーバーの名前を指定します。| トレースするレプリケーション・アラート・モニター・プログラム用のモニター・コントロール・サーバーの名前を指定します。
- -qcap
  - Q キャプチャー・プログラムをトレースすることを指定します。Q キャプチャー・プログラムは -schema パラメーターで識別されます。
- -schema qcapture\_schema
  - トレースする Q キャプチャー・プログラムの名前を指定します。Q キャプチャー・プログラムは指定した Q キャプチャー・スキーマにより識別されます。このパラメーターは -qcap パラメーターと一緒に使用します。
- -qapp
  - Q アプライ・プログラムをトレースすることを指定します。Q アプライ・プログラムは -schema パラメーターで識別されます。
- -schema qapply\_schema
  - トレースする Q アプライ・プログラムの名前を指定します。Q アプライ・プログラムは指定した Q アプライ・スキーマにより識別されます。このパラメーターは -qapp パラメーターと一緒に使用します。
- -cap
  - キャプチャー・プログラムをトレースすることを指定します。キャプチャー・プログラムは -schema パラメーターで識別されます。
- -schema capture\_schema
  - トレースするキャプチャー・プログラムの名前を指定します。キャプチャー・プログラムは指定したキャプチャー・スキーマにより識別されます。このパラメーターは -cap パラメーターと一緒に使用します。
- -app
  - アプライ・プログラムをトレースすることを指定します。アプライ・プログラムは -qualifier パラメーターで識別されます。
- -qualifier apply\_qualifier
  - トレースするアプライ・プログラムの名前を指定します。このアプライ・プログラムは、指定したアプライ修飾子により識別されます。このパラメーターは -app パラメーターと一緒に使用します。



## 解説:

- off
  - 特定の Q キャプチャー・プログラム、Q アプライ・プログラム、キャプチャー・プログラム、アプライ・プログラム、またはレプリケーション・アラート・モニター・プログラムについて、トレース機能をオフにし、使用中の共有メモリー・セグメントを解放することを指定します。
- kill
  - トレース機能を強制的に異常終了させることを指定します。
  - このパラメーターは、何らかの問題により、トレース機能を off パラメーターでオフにできない場合のみ使用してください。
- clr
  - トレース・バッファをクリアすることを指定します。このパラメーターは、トレース・バッファの内容を消去しますが、バッファはアクティブのままにします。
- diag ト
  - レース機能の実行中に、フィルター設定を表示することを指定します。
- resetlock
  - トレース機能のバッファ・ラッチを解放することを指定します。このパラメーターは、エラー状態が起こり、トレース・プログラムがバッファ・ラッチを保留したまま終了した場合に、バッファ・ラッチをエラー状態からリカバリーできるようにします。
- dmp filename
  - トレース・バッファの現在の内容をファイルに書き込むことを指定します。
- -holdlock
  - トレース機能がバッファをコピーするためのメモリーが不足している場合でも、ロックを保留している間に、トレース機能がファイルのダンブまたはコマンドの出力を完了できることを指定します。
- flw
  - トレース機能が作成し、共有メモリーまたはファイルに保管したサマリー情報を表示することを指定します。この情報には、プログラム・フローが含まれ、それぞれの処理およびスレッドごとに、関数と呼び出しのスタック構造がわかるように字下げして表示されます。
- fmt
  - トレース機能が作成し、共有メモリーまたはファイルに保管した詳細情報を表示することを指定します。このパラメーターは、トレースしたデータ構造の内容全体を発生順に表示します。
- v7fmt
  - トレース機能が作成し、共有メモリーまたはファイルに保管した情報を表示することを指定します。このトレース情報はバージョン 7 のフォーマットで表示されます。




## 解説:

- **stat**
  - トレース機能の状況を表示することを指定します。この状況情報には、トレース・バージョン、アプリケーション・バージョン、項目数、バッファ・サイズ、使用中のバッファ量、状況コード、およびプログラム・タイム・スタンプが含まれます。
- **statlong**
  - トレース機能の状況に z/OS バージョン・レベル情報を追加して表示することを指定します。この追加情報には、アプリケーション内の各モジュールのサービス・レベルが含まれ、長ストリングのテキストとして表示されます。
- **-fn filename**
  - ラーリングされたトレース情報を含むファイル名を指定します。ここには、トレース機能からのすべての出力が含まれます。
- **-help**
  - 有効なコマンド・パラメーターを記述と一緒に表示します。
- **-listsymbols**
  - -df パラメーターで使用する有効な関数およびコンポーネント ID を表示します。
- **-b buffer\_size**
  - トレース・バッファのサイズをバイト単位で指定します。数値の後に、キロバイトなら K を、メガバイトなら M を指定できます。これらの文字には大文字小文字の区別はありません。
- **-fs filesize**
  - ミラーリングされたトレース情報ファイルのサイズ制限をバイト単位で指定します。
- **-d diag\_mask**
  - トレース機能により記録されるトレース・レコードのタイプを指定します。トレース・レコードは、以下の診断マスク番号により分類されます。
    - 1 フロー・データ。関数の入口点と出口点が含まれます。
    - 2 基本データ。トレース機能が検出したすべての主要なイベントが含まれます。
    - 3 詳細データ。主要なイベントとその記述が含まれます。
    - 4 パフォーマンス・データ。
  - 重要: 診断マスク番号の大きいものは、診断マスク番号の小さいものを包含していません。
  - これらの番号を 1 つまたは複数入力し、必要なトレース・レコードだけを含む診断マスクを作成することができます。たとえば、-d 4 を指定すると、パフォーマンス・データだけが記録されます。フローとパフォーマンスのデータだけを記録するには -d 1,4 と指定し、すべてのトレース・レコードを記録するには -d 1,2,3,4 (デフォルト) と指定します。番号はコンマで区切ります。
  - トレース機能がグローバル・トレース・レコードを記録しないようにするには、診断マスク番号 0 (ゼロ) を入力します。トレース機能に新しい診断マスク番号を指定する前に、診断レベルをリセットするには、-d 0 を入力します。
- **-df function\_name|component\_name diag\_mask**
  - 特定の関数またはコンポーネント ID をトレースすることを指定します。
  - 関数またはコンポーネント ID 名の後に診断マスク番号 (1、2、3、4) を入力します。1 つまたは複数の番号を入力できます。番号はコンマで区切ります。



DB2 Universal Database

*blank page*




DB2 Universal Database

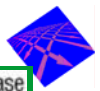
## 不整合は発生していないか

### ■ ASNTDIFF

- ソース表とターゲット表を比較する

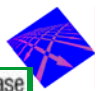
### ■ ASNTREP

- ソース表とターゲット表の差異を修復する



DB2 Universal Database

*blank page*



DB2 Universal Database

## ASNTDIFFユーティリティ

### ■ ソース表とターゲット表の相違を抽出する

1. ソース表にあってターゲット表にないレコード
2. ターゲット表にあってソース表にないレコード
3. 同一キーでデータが異なるレコード

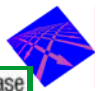
➡ 相違検出表 (DIFF表) に相違を記録する

### ■ SQLレプリケーション(V8)

- ターゲット表のタイプはユーザー・コピー表のみ
  - CCD表、ポイント・イン・タイム表、レプリカ表は不可

### ■ LUW、z/OS(USSおよびJCL)で実行可能

- LUW、z/OS、iSeries上の表を比較
- Federated環境もサポート



DB2 Universal Database

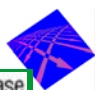
## ASNTDIFFユーティリティ

### ■ ASNTDIFFコマンド

`asntdiff db=server schema=schema where="where_clause"`

#### 【必須のパラメータ】

- `db=server`
  - IBMSNAP\_SUBS\_SET表を含むアプライ・コントロール・サーバー名
- `schema=schema`
  - アプライ・コントロール表のスキーマ名 (ASN)
- `where="where clause"`
  - 比較するソース表とターゲット表を一意に識別できる条件
    - IBMSNAP\_SUBS\_MEMBR表から固有の行を識別できる条件
    - APPLY\_QUAL、SET\_NAME、TARGET\_SCHEMA、TARGET\_TABLE列でユニーク



DB2 Universal Database



## ASNTDIFFユーティリティ

### 【その他のパラメータ】

- `diff_schema=diffable_schema`
  - DIFF表のスキーマ名
  - 省略値はschemaパラメータで指定したスキーマ名
- `diff=diffable`
  - DIFF表名
  - 省略値は“ASNTDIFF”
- `diff_tablespace=diffable_tablespace`
  - DIFF表を作成する表スペース名
  - 指定する表スペースはあらかじめ作成しておく必要がある
- `diff_drop=[y / n]`
  - 既存のDIFF表を再作成するかどうか
  - 省略値は“n”
- `maxdiff=difference_limit`
  - 指定された件数以上の相違があった場合は停止させることができる
  - 省略値は、10,000
- `pwdfile=pwdfile`
  - 各サーバーへ接続するために使用するパスワード・ファイル名
  - 省略値は“asnpwd.aut”
- `diff_path=diff_path`
  - パスワード・ファイルが保管されているパス名
  - 省略値は実行パス



DB2 Universal Database

## ASNTDIFFユーティリティの実行

### ■ パスワード・ファイルの準備

- キャプチャー・コントロール・サーバー
- アプライ・コントロール・サーバー
- ターゲット・サーバー

### ■ コマンド例

- `asntdiff db=mydb schema=asn`  
`where="where set_name='MY_SET' and source_table='T1' "`
  - アプライ・コントロール・サーバー“MYDB”の“ASN”.”IBMSNAP\_SUBS\_MEMBR”表からSUBS\_SET='MY\_SET'かつSOURCE\_TABLE='T1'に基づいたソース表とターゲット表の差異をASN.ASNTDIFF表に抽出



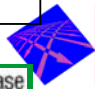
DB2 Universal Database

## 相違検出表(DIFF表)

- ソース表とターゲット表の相違が格納される
  - ASNTDIFFを実行するとソース・サーバー上に作成される
    - 既に存在している場合は既存レコード削除後に相違が格納される
  - 省略時のDIFF表名: ASNTDIFF 表
- 2列以上の列をもつ
  - DIFF列: 相違を示す
  - キー列: レプリケーション・キー列 (複数有)
    - IBMSNAP\_SUBS\_COLS表のIS\_KEY='Y'の列

DIFF	KEY1	KEY2
I 2	00001	01
D 2	00010	02
U 2	00020	02
:	:	:
:	:	:

DIFF表の例

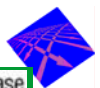


## 相違検出表(DIFF表)

- DIFF列 (CHAR(4)) の値
  - DIFF="I 2"
    - ターゲット表に一致するキーがない
    - ➡ ターゲット表にINSERTが必要
  - DIFF="D 2"
    - ソース表に一致するキーがない
    - ➡ ターゲット表からDELETEが必要
  - DIFF="U 2"
    - ソース表に一致するキーはあるが等しいデータでない
    - ➡ ターゲット表にUPADTEが必要

DIFF	KEY1	KEY2
I 2	00001	01
D 2	00010	02
U 2	00020	02
:	:	:
:	:	:

DIFF表の例



## ASNTDIFFユーティリティの注意点

- 表に更新がない時に実行してください
  - IsolationはURで実行されるため正しい結果が得られません
  - 必要であればCSでバインドしてください
- 実行環境は接続する全てのデータベースとのコード変換がサポートされている必要がある
- キーの重複がある場合は相違点を抽出できない場合があります
  - 次頁参照



DB2 Universal Database

## ASNTDIFFユーティリティの注意点

- キーの重複がある場合は相違点を抽出できない場合があります
  - ソース表で重複しており、ターゲット表に1レコード存在している
  - ソース表に1レコード存在しており、ターゲット表で重複している

実際はソース表とターゲット表間で差異があるが、DIFF表に抽出されないケース

ソース表 キー列 C1	ターゲット表 キー列 C1		DIFF表	
			DIFF列	C1
C1=100 C2='ABC'	C1=100 C2='ABC'	→	OK:レコードなし	
C1=100 C2='ABC'	C1=100 C2='XYZ'	→	U 2	0100
C1=100 C2='ABC'	-	→	I 2	0100
-	C1=100 C2='ABC'	→	D 2	0100
C1=100 C2='ABC'	-	→	I 2	0100
C1=100 C2='ABC'	-	→	I 2	0100
C1=100 C2='ABC'	C1=100 C2='ABC'	→	OK:レコードなし	
-	C1=100 C2='ABC'	→	D 2	0100
-	C1=100 C2='ABC'	→	D 2	0100
C1=100 C2='ABC'	C1=100 C2='ABC'	→	OK:レコードなし	
C1=100 C2='ABC'	C1=100 C2='ABC'	→	OK:レコードなし	



DB2 Universal Database

## ASNTDIFFユーティリティの注意点(つづき)

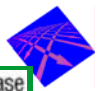
### ■ キー情報をシステム一時ファイルに保管

- 実行するシステム上に作成
  - AIX: /tmp
  - Windows: ユーザー環境変数 "TMP" に指定されたパス

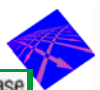
### ■ ASNTDIFF表へのレコードのInsertは、1UOWで行うため、アクティビティ・ログフルの可能性がある

- ただし、ログフルが発生するほどDIFF表へのINSERTが多いことは、相違件数が大量にあることをいみするため、その場合はフルリフレッシュを検討する

### ■ UTF-8、UTF-16には対応していない



*blank page*



## ASNTREPユーティリティ

### ■ソース表とターゲット表の差異を自動修復する

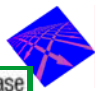
- DIFF表をもとにターゲット表を更新する
  - ASNTDIFFを実行しておく必要がある
- 「DIFF」列
  - “I 2” はターゲット表にレコード挿入
  - “U 2” はターゲット表のレコード更新
  - “D 2” はターゲット表のレコード削除

### ■サポートするターゲットのタイプ

- ASNTDIFFがサポートするタイプと同じ

### ■Linux,Unix,Aixで実行可能

- (LUW、z/OS、iSeries上の表)
- (Federated環境もサポート)



DB2 Universal Database

## ASNTREPユーティリティ

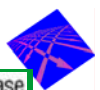
### ■ASNTREPコマンド

```
asntrep db=server schema=schema where="where_clause"
```

#### 【必須のパラメータ】

- db=*server*
  - IBMSNAP\_SUBS\_SET表を含むアプライ・コントロール・サーバー名
- schema=*schema*
  - アプライ・コントロール表のスキーマ名 (ASN)
- where="*where\_clause*"
  - 比較するソース表とターゲット表を一意に識別できる条件
    - IBMSNAP\_SUBS\_MEMBR表から固有の行を識別できる条件
    - APPLY\_QUAL、SET\_NAME、TARGET\_SCHEMA、TARGET\_TABLE列でユニーク

ASNTDIFFコマンドと同じ



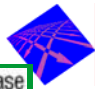
DB2 Universal Database

## 解説:

### 【その他のパラメータ】

- `diff_schema=difftable_schema`
  - DIFF表のスキーマ名
  - 省略値はschemaパラメータで指定したスキーマ名
- `diff=difftable`
  - DIFF表名
  - 省略値は“ASNTDIFF”
- `diff_tablespace=difftable_tablespace`
  - DIFF表を作成する表スペース名
  - 指定する表スペースはあらかじめ作成しておく必要がある
- `diff_drop=[y / n]`
  - 既存のDIFF表を再作成するかどうか
  - 省略値は“n”
- `pwdfile=pwdfile`
  - 各サーバーへ接続するために使用するパスワード・ファイル名
  - 省略値は“asnpwd.aut”
- `diff_path=diff_path`
  - パスワード・ファイルが保管されているパス名
  - 省略値は実行パス

ASNTDIFFコマンドと同じ



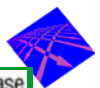
DB2 Universal Database

## ASNTREPユーティリティを実行する

### ■ ASNTDIFFを実行しDIFF表を用意する

### ■ ASNTREPを実行する

- ① DIFF表とソース表からDIFFコピー表がターゲット・サーバーに作成される
- ② ターゲット表を更新
  - DIFF列が' I 2' のレコードがターゲットにINSERT
  - DIFF列が' U 2' のレコードがターゲットにUPDATE
  - DIFF列が' D 2' のレコードがターゲットにDELETE



DB2 Universal Database

## ASNTREPユーティリティを実行する

### ソース・サーバー

#### DIFF表

DIFF	KEY1	KEY2
I 2	00001	01
D 2	00010	02
U 2	00020	02
:	:	:
:	:	:

KEY1	KEY2	列
00001	01	ABC
00020	02	NEW
:	:	:
:	:	:

ソース表

### ターゲット・サーバー

#### DIFFコピー 表

(DIFF表とソース表を元に作成される)

DIFF	KEY1	KEY2	列
I 2	00001	01	ABC
D 2	00010	02	
U 2	00020	02	NEW
:	:	:	:
:	:	:	:

KEY1	KEY2	列
00001	01	ABC
<del>00010</del>	<del>02</del>	<del>ABC</del>
00020	02	OLD=>NEW
:	:	:
:	:	:

ターゲット表

INSERT

DELETE

UPDATE

②

①



## ASNTREPユーティリティの注意点

- ターゲット表の属性を考慮
  - ターゲット表のユニーク制約のために修復が失敗することがある
  - ターゲット表の参照保全制約のために修復が失敗することがある
- ASNTREP実行中のエラーは全ての処理をロールバックする
- 相違のレコード数によっては、フルリフレッシュやロードを実施して下さい
- DIFFコピー表は手動で削除する必要がある

DIFF表の例



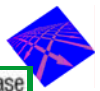


## ASNANALYZEユーティリティ

### ■レプリケーション環境の状況レポートを生成する

- SQLレプリケーション・コントロール表を分析する
- 問題分析時に使用
- 出力をIBMソフトウェア・サポートに提出する

### ■Linux、Unix、Windowsから実行



DB2 Universal Database

## ASNANALYZEユーティリティ

### ■コマンド

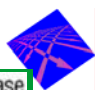
- `asnanalyze -db db_alias`

### ■【必須パラメータ】

- `-db db_alias`
  - キャプチャー・コントロール・サーバーまたは、ターゲット・サーバーまたは、アプライ・コントロール・サーバーを指定

### ■使用例

- `asnanalyze -db TEST`
  - TESTデータベースのコントロール表をレポートする



DB2 Universal Database

## 解説:

### 【その他のパラメータ】

- **-la *level of analysis***
  - レポートする分析のレベルを指定
    - standard (省略値)
    - detailed
    - simple
- **-tl *n***
  - APPLYTRAIL 表から検索する項目の日付範囲を指定(0～30日)
  - 省略値は3日
- **-at *n***
  - APPLYTRACE 表から検索する項目の日付範囲を指定(0～30日)
  - 省略値は3日
- **-ct *n***
  - CAPTRACE表から検索する項目の日付範囲を指定(0～30日)
  - 省略値は3日
- **-cm *n***
  - CAPMON 表から検索する項目の日付範囲を指定(0～30日)
  - 省略値は3日
- **-sg *n***
  - SIGNAL表から検索する項目の日付範囲を指定(0～30日)
  - 省略値は3日
- **-aq *apply\_qualifier***
  - アプライ修飾子
- **-cs *capture\_schema***
  - キャプチャー・スキーマ名



DB2 Universal Database

## 解説

- **-od *output\_directory***
  - アナライザー・レポートを保管するディレクトリーを指定
  - 省略値は現行ディレクトリ
- **-fn *output\_filename***
  - アナライザー・レポート出力ファイルの名前を指定
  - ファイル名がすでに存在する場合、ファイルは上書きされる
  - デフォルトのファイル名は、asnanalyze.htm
- **-pw *password\_filepath***
  - パスワード・ファイルの名前とパスを指定
  - 省略した場合アナライザーは現行ディレクトリーでasnpwd.autファイルを探す。



DB2 Universal Database

## 解説:

## ■ASNANALYZEレポート出力結果



## ASNANALYZEユーティリティ

- iSeriesではANZDPRコマンドが使用できる
  - ASNANALYZEユーティリティと同等の機能を持つ
- 使用例
  - ANZDPR RDB(\*LOCAL MYDB)



## 解説:

### ■ ANZDPRパラメーター

- **RDB(*ldb-name*)**
  - 分析対象のデータベースを指定
  - 省略値は\*LOCAL
    - ローカル・システム上のデータベース。
  - *ldb-name*
    - データベースを示す、RDB ディレクトリー項目名。
- **OUTFILE(*library-name file-name*)**
  - アナライザー出力の保管に使用されるライブラリーとファイル名を指定
  - このコマンドは、出力を HTML ファイルに書き込む
  - *library-name*の省略値は\*CURLIB
    - 現行ライブラリー
  - *library-name*
    - ライブラリーの名前。
  - *file-name*の省略値はANZDPR
    - 出力は、ANZDPR という名前の HTML ファイルに書き込む
  - *file-name*
    - HTML 出力ファイルの名前。
    - ファイル名がすでに存在する場合、ファイルは上書きされる
    - ファイル名が存在しない場合は、RCDLEN(512) および SIZE(\*NOMAX) という属性のファイルがコマンドにより作成される
- **ANZLVL**
  - 報告される分析のレベルを指定
    - \*STANDARD (省略値)
    - \*SIMPLE
    - \*DETAILED



DB2 Universal Database

## 解説:

- **CAPTRC *no-of-days***
  - CAPTRACE表から報告される項目の日付範囲を指定(0~30日)
  - 省略値は3
- **APYTRC *no-of-days***
  - APPLYTRACE表から報告される項目の日付範囲を指定(0~30日)
  - 省略値は3
- **APYTRAIL *no-of-days***
  - APPLYTRAIL表から報告される項目の日付範囲を指定(0~30日)
  - 省略値は3
- **SIGTBL *no-of-days***
  - SIGNAL表から報告される項目の日付範囲を指定(0~30日)
  - 省略値は3
- **CAPMON *no-of-days***
  - CAPMON表から報告される項目の日付範囲を指定(0~30日)
  - 省略値は3
- **APYQUAL *apply-qualifier***
  - 分析対象のアプライ修飾子を指定
  - 省略値は\*ALL
  - 分析対象のアプライ修飾子の名前。最大 10 のアプライ修飾子を入力できます。
- **CAPCTLLIB *library-name***
  - キャプチャー・コントロール・ライブラリーの、キャプチャー・スキーマを指定
  - 省略値は\*ALL1



DB2 Universal Database

## レプリケーション・センターでのモニタリング

### ■ キャプチャー

- スループット分析の表示
- 待ち時間表示
- 状況の照会

### ■ アプライ

- スループット分析の表示
- エンド ツー エンド待ち時間表示
- 状況の照会



*blank page*

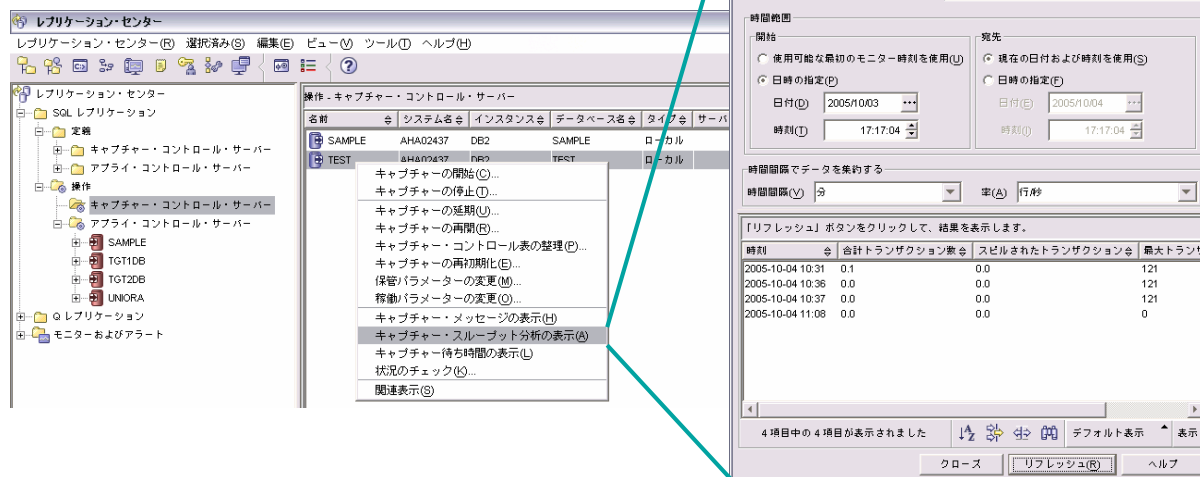


## キャプチャー:スループット分析の表示

### ■ スループット分析の表示

- ログから挿入、あるいはスキップされた行数
- CD表から整理(プルーニング)された行数
- コミットされたトランザクション数
- メモリー使用量

### ■ データを時間間隔で集約して表示することも可能



## 解説:

- キャプチャー・プログラムについてパフォーマンス統計を表示します。IBMSNAP\_CAPMON表からの情報を表示し、特定のタスクが実行された速度を計算することができます。
- 各タイプの情報が提供するのは、キャプチャー・プログラムのパフォーマンスの一部でしかなく、全体を把握するには、「キャプチャー待ち時間」ウィンドウと同じく、4つのすべての異なる情報タイプについて調べる必要があります。
- 指定した時間の範囲でパフォーマンスを分析することができます。範囲は、キャプチャー・プログラムが開始した時刻から、またはユーザーが指定した時刻から開始させます。範囲の終了は、現在の日時、またはユーザーが指定する時刻までとします。
- ログから挿入された、あるいはスキップされた行数、CD表から整理された行数、またはコミットされたトランザクション数を表示するよう選択した場合、時間の範囲を1秒、1分、1時間、1日、1週間のインターバルに分割することができます。たとえば、1分のインターバルを選択すると、時間の範囲は分刻みで分割され、1つの行が分ごとに表示されます。表示された各行は、その1分の間のアクティビティをサマリーしています。
- また、異なる比率に応じて情報を平均させることもできます。たとえば、CD表から整理した行数を表示させるとします。情報を1分のインターバルで表示し、整理された行数を秒単位で表示するよう選択することができます。
- 次の表に、データの集計用のオプションをまとめてあります。ログから挿入された、あるいはスキップされた行数、CD表から整理された行数、またはコミットされたトランザクション数を表示するよう選択した場合にのみ、これらのオプションを利用することができます。

時間インターバル	比率	データ表示の形式
時間インターバルなし	N/A	ローデータ
秒、分、時間、日、または週	なし	秒、分、時間、日、または週ごとに1行。各行で、その時間インターバルごとのデータを合計しています。
秒、分、時間、日、または週	ログから挿入された、あるいはスキップされた行数、CD表から整理された行数のどちらかを表示する場合、次のようになります。 行/秒 行/分 行/時間 コミットされたトランザクション数を表示する場合、次のようになります。 トランザクション/秒 トランザクション/分 トランザクション/時間 トランザクション/日	秒、分、時間、日、または週ごとに1行。各行で、選択済みの比率に応じて平均を表示しています。

## キャプチャー:スループット分析の表示

### ■ ログから挿入、あるいはスキップされた行数

キャプチャー・スループット分析

AHA02437 - DB2 - TEST

キャプチャー・スキーマ(H)

使用可能な最初のモニター時刻を使用(U)

時間範囲

開始

☐ 使用可能な最初のモニター時刻を使用(U)

☒ 日時の指定(P)

日付(D)

時刻(T)

宛先

☒ 現在の日付および時刻を使用(S)

☐ 日時の指定(P)

日付(E)

時刻(I)

時間間隔でデータを集約する

時間間隔(V)

率(A)

「リフレッシュ」ボタンをクリックして、結果を表示します。

時刻	挿入行数	再キャプチャー機能	トリガーによってス
2005-10-04 10:31:25.125001	2.0	0.0	0.0
2005-10-04 10:36:26.118000	0.0	0.0	0.0
2005-10-04 10:37:50.289000	0.0	0.0	0.0
2005-10-04 11:08:41.140000	0.0	0.0	0.0

4 項目中の 4 項目が表示されました

クローズ リフレッシュ(R) ヘルプ



## 解説:

### ■ 「ログから挿入された、あるいはスキップされた行数」

- キャプチャー・プログラムによって CD 表に挿入された行数を表示、および、そのインターバルを通して再キャプチャー機能、ユーザー定義のトリガー、および CHGONLY 機能にしたがって挿入されなかった行数を表示します。

### ■ 結果の列

列	記述
時刻	指定したインターバルが開始した時刻。時間インターバルを選択していない場合、モニター・インターバルの開始のタイム・スタンプです。
挿入行数	そのインターバルを通して、すべてのレプリケーション・ソースについてキャプチャー・プログラムが CD 表に挿入した行数。
再キャプチャー機能にしたがってスキップされた行数	そのインターバルを通して、キャプチャー・プログラムが処理したが、再キャプチャー機能により挿入されなかった行数。
トリガーにしたがってスキップされた行数	そのインターバルを通して、キャプチャー・プログラムが処理したが、ユーザー定義の抑止トリガーにより挿入されなかった行数。
CHGONLY にしたがってスキップされた行数	そのインターバルを通して、キャプチャー・プログラムが処理したが、CHGONLY 機能により挿入されなかった行数。





## キャプチャー:スループット分析の表示

### ■ CD表から整理(ブルーニング)された行数

時刻	整理された行数
2005-10-04 10:31:25.295000	54
2005-10-04 10:31:25.305000	4
2005-10-04 10:31:25.385000	323



## 解説:

### ■ CD 表から整理された行数

- そのインターバルを通してキャプチャー・プログラムが複製したすべてのソース表のうち CD 表から整理された行数を表示します。

### ■ 結果の列

列	記述
時刻	指定したインターバルが開始した時刻。時間インターバルを選択していない場合、モニター・インターバルの開始のタイム・スタンプです
整理された行数	そのインターバルを通して、すべてのレプリケーション・ソースについて CD 表から整理された行数。



## キャプチャー:スループット分析の表示

### ■コミットされたトランザクション数

時刻	合計トランザ...	スピルされたトラン...	最大トランザクショ
2005-10-04 10:31:25.125001	6.0	0.0	121
2005-10-04 10:36:26.118000	0.0	0.0	121
2005-10-04 10:37:50.289000	0.0	0.0	121
2005-10-04 11:08:41.140000	0.0	0.0	0



## 解説:

- 「コミットされたトランザクション数」
  - そのインターバルを通してコミットされたトランザクションの合計、予備としてディスクに入れられたトランザクションの合計、および最大のトランザクションのサイズを表示します。
- 結果の列

列	記述
時刻	指定したインターバルが開始した時刻。時間インターバルを選択していない場合、モニター・インターバルの開始のタイム・スタンプです。
合計トランザクション数	そのインターバルを通してキャプチャーされた、レプリケーション・ソースでのトランザクション数。
スピルされたトランザクション	メモリー制限のため、そのインターバルを通してハード・ディスクにスピルされた、レプリケーション・ソースでのトランザクション数。
最大トランザクション・サイズ	そのインターバルを通して発生した、レプリケーション・ソースで最大のトランザクション。このサイズを認識しておく、キャプチャー・プログラムに使用可能なメモリーを増やすかどうかを決定するのに役立ちます。



## キャプチャー:スループット分析の表示

### ■メモリ使用量



## 解説:

### ■「メモリ使用量」

- キャプチャー・プログラムが使用したメモリの合計値、およびキャプチャー・プログラムが使用するメモリ使用量が、キャプチャー・モニター (IBMSNAP\_CAPMON) 表にある MEMORY\_LIMIT パラメーターに指定した値を超える可能性のある使用量を表示します。(このパラメーターの値を、キャプチャー・プログラムのための「稼働パラメーターの変更」ウィンドウで変更してある場合、新しい値はキャプチャー・モニター (IBMSNAP\_CAPMON) 表に反映されません。)

### ■ 結果の列

列	記述
時刻	指定したインターバルが開始した時刻。時間インターバルを選択していない場合、モニター・インターバルの開始のタイム・スタンプです
使用済みメモリ (MB)	キャプチャー・プログラムが使用したメモリの合計
超過したメモリ (MB)	キャプチャー・プログラムが、MEMORY_LIMIT に指定された以上のメモリを使用すると、超過して使用したメモリの量をこの列に表示します。



## キャプチャー: 待ち時間表示

キャプチャー待ち時間

AHA02437 - DB2 - TEST

キャプチャー・スキーマ(H) [ASN]

時間範囲

開始

☐ 使用可能な最初のモニター時刻を使用(U)

☐ 現在の日および時刻を使用(S)

日時指定(D)

日付(D) 2005/10/03

時刻(T) 17:48:16

時間範囲でデータを集約する

時間間隔(V) インターバルなし

時刻

時刻	待ち時間
2005-10-04 10:31:25.0001	29
2005-10-04 10:36:26.118000	9
2005-10-04 10:37:50.289000	29
2005-10-04 11:08:41.140000	1
2005-10-04 17:43:35.231000	24

5項目中の5項目が表示されました

クローズ リフレッシュ

レプリケーション・センター

レプリケーション・センター(R) 選択済み(S) 編集(E) ビュー(V) ツール(T) ヘルプ(H)

レプリケーション・センター

SQL レプリケーション

定義

- キャプチャー・コントロール・サーバー
- アプライ・コントロール・サーバー
- 操作
- キャプチャー・コントロール・サーバー
- アプライ・コントロール・サーバー
- SAMPLE
- TG1DB

操作 - キャプチャー・コントロール・サーバー

名前	システム名	インスタンス	データベース名	タイプ	サ
SAMPLE	AHA02437	DB2	SAMPLE	ローカル	
TEST	AHA02437	DB2	TEST	ローカル	

キャプチャーの開始(C)...

キャプチャーの停止(T)...

キャプチャーの延期(D)...

キャプチャーの再開(R)...

キャプチャー・コントロール表の整理(P)...

キャプチャーの再初期化(R)...

保管パラメーターの変更(M)...

稼働パラメーターの変更(O)...

キャプチャー・メッセージの表示(M)

キャプチャー・スループット分析の表示(A)

キャプチャー待ち時間の表示(L)

状況のチェック(C)...

関連表示(S)

時間間隔を選択した場合



## 解説:

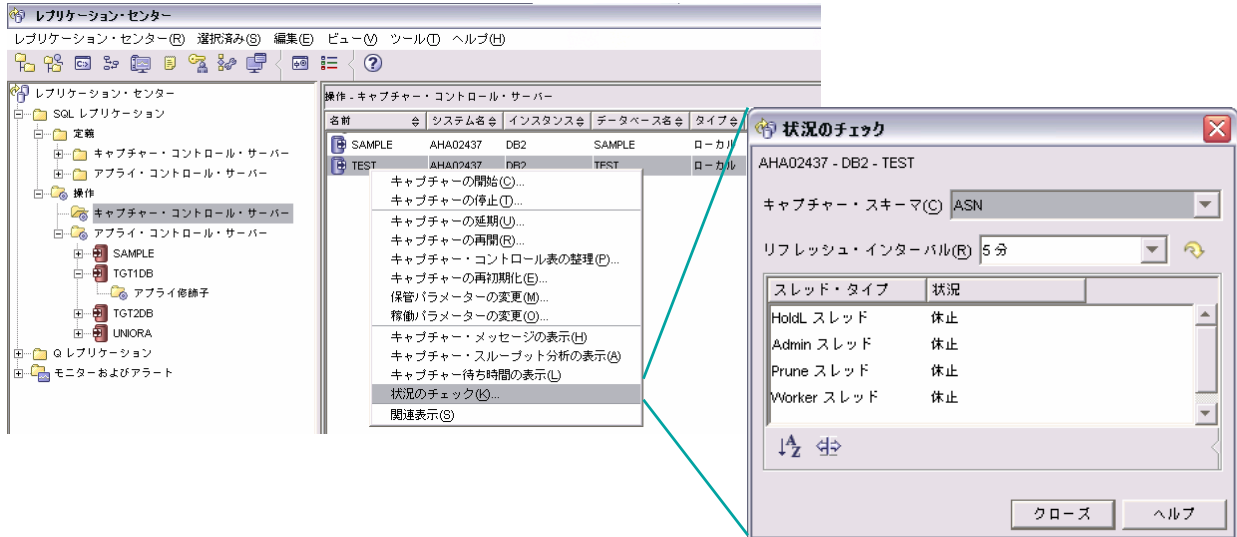
- 「キャプチャー待ち時間」
  - ソースでデータが更新されてから、キャプチャー・プログラムによって取り込まれるまでのおよその時間を表示します。経過時間は一定期間にわたり、CD表の中のデータの新鮮さを示します。この平均待ち時間は、キャプチャー・キャプチャー・モニター (IBMSNAP\_CAPMON) 表の情報から生成できます。この情報の派生元は、登録 (IBMSNAP\_REGISTER) 表です。
- 現行のキャプチャー待ち時間は登録 (IBMSNAP\_REGISTER) 表の中のグローバル・レコードからの SYNCHTIME 列と CURRENT\_TIMESTAMP 値を使用して計算されます。
  - $(CURRENT\_TIMESTAMP) - (SYNCHTIME)$
- キャプチャー待ち時間は時間とともに変化し、これらの変更のヒストリーはキャプチャー・モニター (IBMSNAP\_CAPMON) 表に保管されます。レプリケーション・センターはキャプチャー・モニター表の中の情報を使用して、平均待ち時間、またはヒストリー待ち時間を計算します。平均待ち時間の場合の公式は、現行待ち時間のものと同じですが、CURRENT\_TIMESTAMP 値でなく、MONITOR\_TIME 値が使用されます。
- MINITOR\_TIME 値は、キャプチャー・プログラムによってキャプチャー・モニター表に行が挿入された時間を示すタイムスタンプです。平均待ち時間は、秒、分、時間、日、または週で表示できます。



## キャプチャー: 状況の照会

### ■ キャプチャー・スレッド状況の照会

- メイン・スレッド以外のスレッドの状況を照会
- 自動的リフレッシュが可能



## 解説:

- キャプチャー・コントロール・サーバーで実行中のキャプチャー・プログラムの現在の状況を調べます。各スレッドの状態を記述するメッセージが送られてきます。
- データベース・サーバーは UNIX、Windows、または z/OS 上で実行されている必要があります。
- キャプチャー・プログラムには、管理スレッド、ブルーニング・スレッド、ワーカー・スレッド、およびシリアルライゼーション・スレッドの 4つのスレッドがあります。
- プログラムが正しく機能しているかどうかを、メッセージを受け取ることによって認識できます。一般的には、ワーカー・スレッド、管理スレッド、およびブルーニング・スレッドは作業状態にあります。スレッドが作業状態にある時には、実行しなければならないタスクを実行しています。たとえば、ブルーニング・スレッドは、CD 表と作業単位 (IBMSNAP UOW) 表を整理しています。また、シリアルライゼーション・スレッドは一般的に、待ち状態にあります。
- スレッドの状態
  - 作業中 (Is doing work)
    - 処理中です。
  - 存在します (Exists)
    - スレッドは存在するが、開始できない。IBM サービス・サポートにご連絡ください。
  - 開始済み (Was started)
    - スレッドは開始済みだが、初期化できない。スレッドが多すぎる、CPU が不足している、などのシステム・リソース上の問題がないか調べてください。
  - 初期化中 (Is initializing)
    - スレッドは初期化されるが、機能できない。IBM サービス・サポートにご連絡ください。
  - 休止中 (Is resting)
    - この状態は、キャプチャー・プログラムのスレッドだけに関係する。スレッドがこの状態にある時には、キャプチャー・プログラムが中断されており、操作の再開を待っている状態です。
  - 停止済み (Is stopped)
    - スレッドは実行されていません。スレッドが停止した理由を示すメッセージがないか、アプライ・トレール (IBMSNAP APPLYTRAIL) またはキャプチャー・トレース (IBMSNAP CAPTRACE) 表を調べてください。たとえば、スレッドのブルーニングが停止したことを示すメッセージを受け取った場合は、その理由を確認するために IBMSNAP CAPTRACE 表を調べてください。表が大きすぎて、すぐに整理したい場合は、キャプチャー・プログラムを停止し、ブルーニング・スレッドを開始するためにもう一度キャプチャー・プログラムを開始することができます。

## レプリケーション・センターでのモニタリング

### ■ キャプチャー

- スループット分析の表示
- 待ち時間表示
- 状況の照会

### ■ アプライ

- スループット分析の表示
- エンド ツー エンド待ち時間表示
- 状況の照会



*blank page*



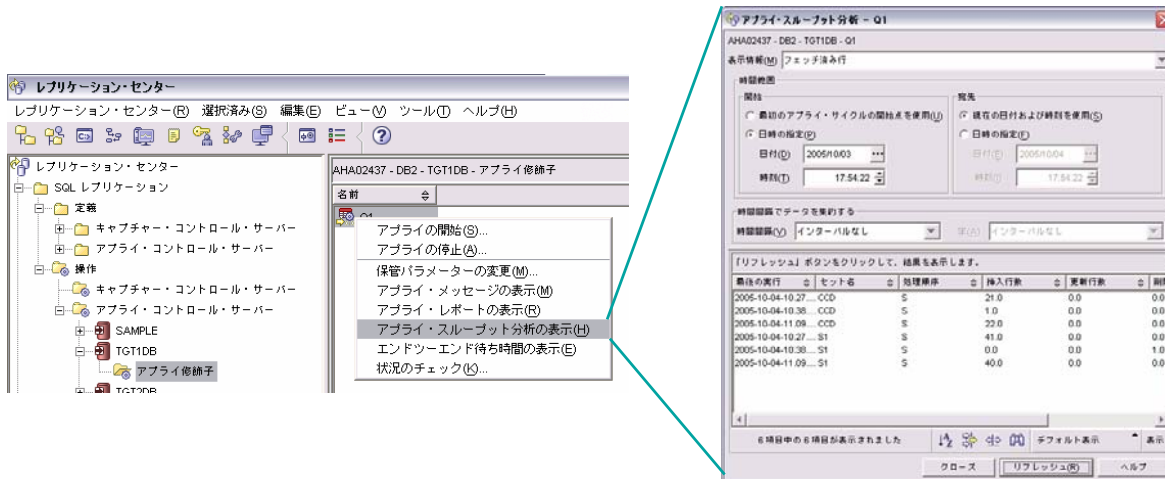


## アプライ:スループット分析の表示

### ■ 分析対象

- フェッチ済み行
- 経過時間

### ■ データを時間間隔で集約して表示することも可能



## 解説:

- 「アプライ・スループット分析」
  - アプライ修飾子についてパフォーマンス統計を表示します。アプライ・修飾子によって処理されたサブスクリプション・セットにあるターゲット表に挿入された行、更新された行、再加工された 行の数を確認することができます。また、アプライ修飾子とそのサブスクリプション・セットの処理にかかった時間も確認できます。
- 重要: このダイアログは、IBMSNAP\_APPLYTRAILコントロール表から情報を検索します。このダイアログを使用するようにする場合は、IBMSNAP\_APPLYTRAIL表のうちアプライ・プログラムのスループットの分析を表示するための時間の範囲で挿入された行から整理しないでください。
- アプライ・プログラムのパフォーマンスを分析するための時間の範囲を選択します。範囲は、アプライ・プログラムが開始した時刻から、またはユーザーが指定した時刻から開始させます。範囲の終了は、現在の日時、またはユーザーが指定する時刻までとします。未来の日時を指定すると、レプリケーション・センターは現在の日時を設定します。
- 時間の範囲は、1 秒、1 分、1 時間、1 週間のインターバルで表示させることができます。たとえば、1 分のインターバルを選択すると、時間の範囲は分刻みで分割され、1 つの行が分ごとに表示されます。表示された各行は、その 1 分の間のアクティビティをサマリーしています。そのアクティビティはさらに、異なる比率でサマリーすることができます。たとえば、CD 表から整理した行数を表示させるとします。情報を 1 分のインターバルで表示し、整理された行数を秒単位または分単位で表示するよう選択することができます。
- 時間の範囲をインターバルで分割しないよう選択すると、「アプライ・スループット分析」ウィンドウには IBMSNAP\_APPLYTRAIL表に挿入された通りにデータを表示します。



## アプライ:スループット分析の表示

### ■ フェッチ済み行

アプライ:スループット分析 - Q1

表示情報(M) フェッチ済み行

時間範囲

開始

☐ 最初のアプライ・サイクルの開始点を使用(U)

☒ 日時の指定(P)

日付(D) 2005/10/03

時刻(T) 17:54:22

宛先

☒ 現在の日付および時刻を使用(S)

☐ 日時の指定(F)

日付(E) 2005/10/04

時刻(I) 17:54:22

時間範囲でデータを集約する

時間範囲(Y) インターバルなし

率(A) インターバルなし

「リフレッシュ」ボタンをクリックして、結果を表示します。

最後の実行	セット名	処理順序	挿入行数	更新行数	削除
2005-10-04-10.27.... CCD		S	21.0	0.0	0.0
2005-10-04-10.38.... CCD		S	1.0	0.0	0.0
2005-10-04-11.09.... CCD		S	22.0	0.0	0.0
2005-10-04-10.27.... S1		S	41.0	0.0	0.0
2005-10-04-10.38.... S1		S	0.0	0.0	1.0
2005-10-04-11.09.... S1		S	40.0	0.0	0.0

6 項目中の 6 項目が表示されました

デフォルト表示

表示

クローズ リフレッシュ(R) ヘルプ



DB2 Universal Database

## 解説:

### ■ 「フェッチ済み行」

- ターゲット表で挿入された行、更新された行、削除された行、および再加工された行数を表示

### ■ 結果の列

列	記述
最後の実行	Applytrail (IBMSNAP_APPLYTRAIL) 表の LASTRUN 列の値。
セット名	サブスクリプション・セットの名前
処理順序	Update-anywhere レプリケーションのシナリオでの、処理の順序を表す値。
挿入行数	サブスクリプション・サイクル中にサブスクリプション・セット・メンバーに挿入された行数。
削除行数	サブスクリプション・サイクル中にサブスクリプション・セット・メンバーから削除された行数。
更新行数	サブスクリプション・サイクル中にサブスクリプション・セット・メンバーから更新された行数。
再加工された行数	サブスクリプション・サイクル中にサブスクリプション・セット・メンバーで再加工された行数。次の 2 つの条件のいずれかの場合に、アプライ・プログラムは変更を再加工します。 一行が既にターゲット表に存在するため挿入が失敗するときは、アプライ・プログラムは挿入を既存行の更新に変換します。 一行がターゲット表に存在しないために更新が失敗するときは、アプライ・プログラムは更新を挿入に変換します。



DB2 Universal Database

## アプライ:スループット分析の表示

### ■ 経過時間

アプライ:スループット分析 - Q1

AHA02437 - DB2 - TGT1DB - Q1

表示情報(M) 経過時間

時間範囲

開始

☐ 最初のアプライ・サイクルの開始点を使用(U)

☒ 現在の日付および時刻を使用(S)

日付(D) 2005/10/03

時刻(T) 17:54:22

宛先

☒ 現在の日付および時刻を使用(S)

☐ 日時の指定(F)

日付(E) 2005/10/04

時刻(I) 17:54:22

時間間隔でデータを集約する

時間間隔(V) インターバルなし

「リフレッシュ」ボタンをクリックして、結果を表示します。

セット名	処理順序	時刻	経過時間
CCD	S	2005-10-04 10:27:04.440001	0
CCD	S	2005-10-04 10:38:26.581000	1
CCD	S	2005-10-04 11:09:20.978000	0
S1	S	2005-10-04 10:27:01.887002	2
S1	S	2005-10-04 10:38:24.799000	1
S1	S	2005-10-04 11:09:20.247000	0

6 項目中の 6 項目が表示されました

デフォルト表示\*

表示

クローズ リフレッシュ(R) ヘルプ



## 解説:

### ■ 「経過時間」

- サブスクリプション・セットの処理にかかった時間を表示します。(単位: 秒)
- (SOURCE\_CONN\_TIME - ENDTIME)

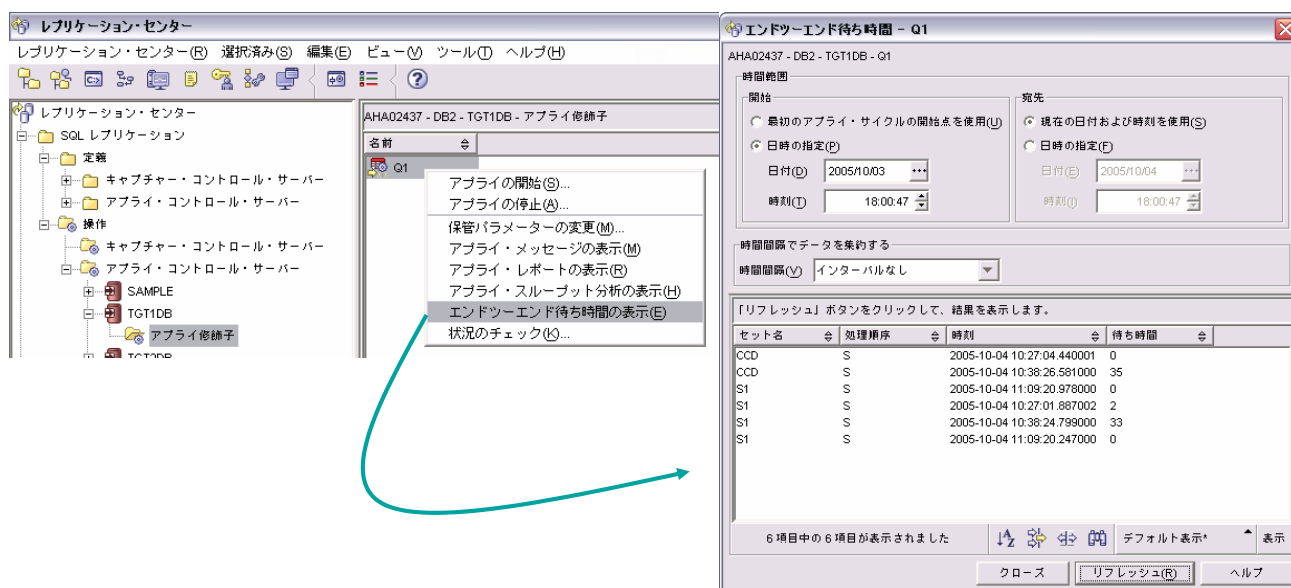
### ■ 結果の列

列	記述
セット名	サブスクリプション・セットの名前
処理順序	Update-anywhere レプリケーションのシナリオでの、処理の順序を表す値。
時刻	時間のインターバルの開始。
経過時間	時間インターバルでサブスクリプション・セットの処理にかかった経過時間。



## アプライ:エンド ツー エンド待ち時間表示

### ■トランザクションの複製にかかる時間の平均を表示



## 解説:

- 「エンド ツー エンド待ち時間の表示」
  - キャプチャー・プロセスおよびアプライ・プロセスが、サブスクリプション・セット内部であらゆるトランザクションを差分レプリケーションで複製するためにかかった時間の平均を示す、およびその値を表示します。
  - 時間の範囲での各アプライ・サイクルごとのサブスクリプション・セットについておおよその待ち時間を「エンドツーエンド待ち時間」ウィンドウに表示します。また、あるインターバルに時間を分割し、そのインターバルごとの平均待ち時間、最大待ち時間、および最少待ち時間を表示することができます。
- 一定期間のアプライ・サイクルごとに、サブスクリプション・セットのおおよその待ち時間を表示できます。また時間をインターバルに分けて、インターバルごとの平均待ち時間を表示することもできます。
- レプリケーション・センターでは次の式を使用してエンド ツー エンド待ち時間を計算します。
  - $(\text{ENDTIME} - \text{LASTRUN}) + (\text{SOURCE\_CONN\_TIME} - \text{SYNCHTIME})$ 
    - ENDTIME アプライがサブスクリプション・セットの処理を終了した時間
    - LASTRUN アプライがサブスクリプション・セットの処理を開始した時間
    - SOURCE\_CONN\_TIME アプライがデータをフェッチするためにキャプチャー・コントロール・サーバーに接続した時間
    - SYNCHTIME キャプチャーによるCD表へのデータのコミットの最新の時刻

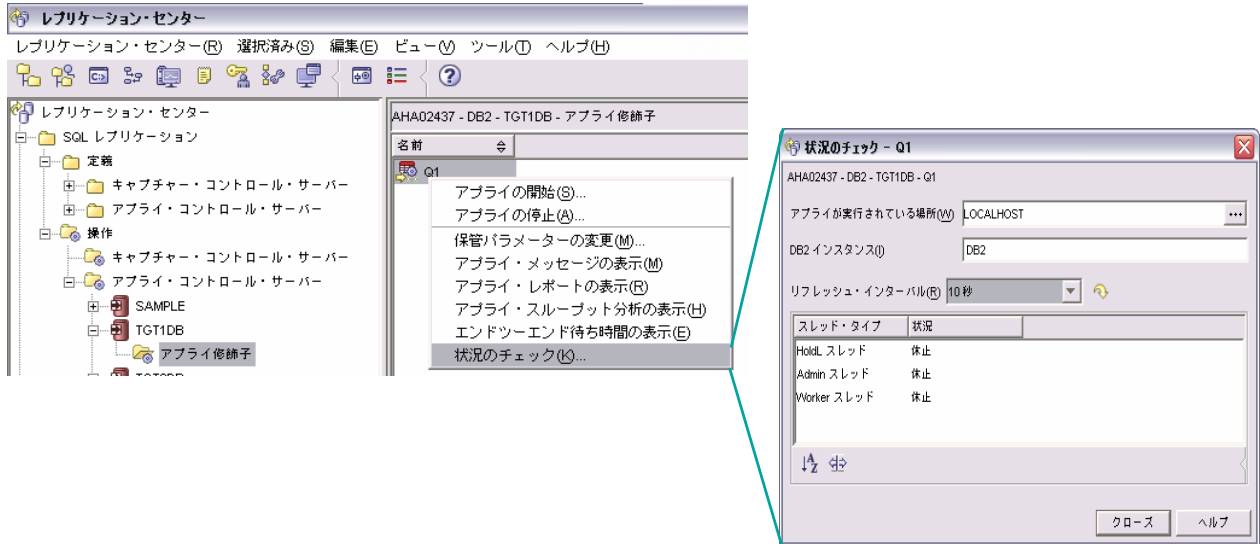
#### ■ 結果の列

列	記述
セット名	サブスクリプション・セットの名前
処理順序	Update-anywhere レプリケーションのシナリオでの、処理の順序を表す値。
時刻	時間インターバルを選択しなかった場合、サブスクリプション・セットについて最後にアプライ・サイクルが開始した時刻を、この列に表示します。時間インターバルを選択した場合、この列にはいずれかのインターバルの開始時刻を表示します。
待ち時間または平均待ち時間	時間インターバルを選択しなかった場合、1つのアプライ・サイクルについて待ち時間を秒でこの列に表示します。時間インターバルを選択した場合、1つのインターバルについて平均待ち時間を秒でこの列に表示します。
最大待ち時間	(この列は、時間インターバルを選択してある場合にのみ表示されます。) そのインターバルにおける、最大待ち時間 (秒)
最小待ち時間	(この列は、時間インターバルを選択してある場合にのみ表示されます。) そのインターバルにおける、最少待ち時間 (秒)

## アプライ: 状況の照会

### ■アプライ スレッド状況の照会

- メイン・スレッド以外のスレッドの状況を照会
- 自動的リフレッシュが可能



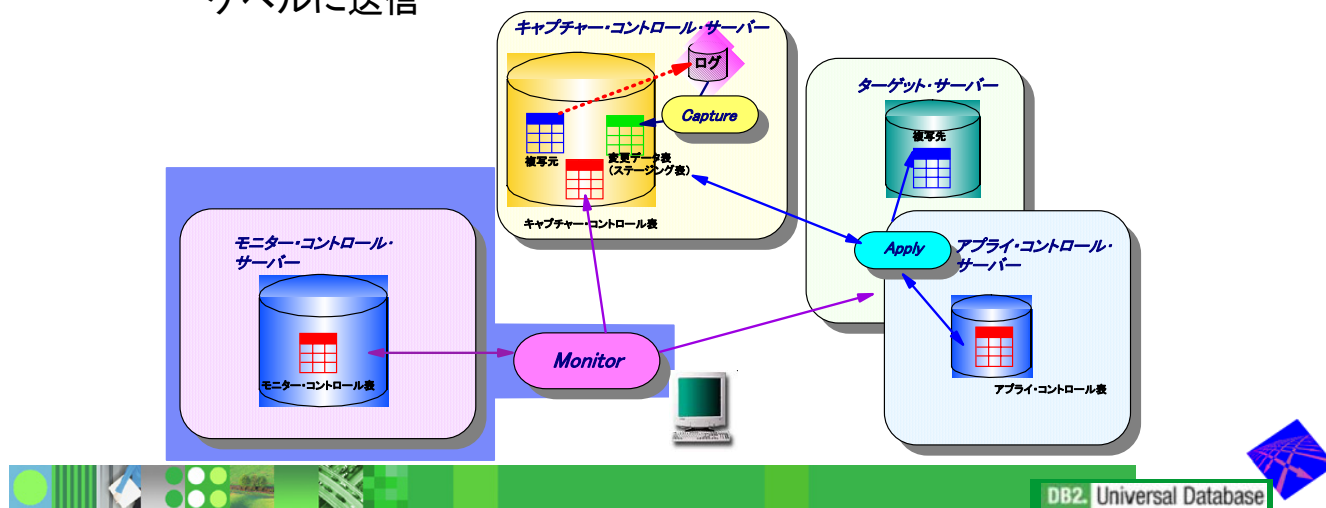
## 解説:

- アプライ・コントロール・サーバー上にコントロール表があるアプライ・プログラムの現在の状況を調べます。これにより、スレッドの状態を記述するメッセージが送られてきます。
- データベース・サーバーは UNIX、Windows、または z/OS 上で実行されている必要があります。
- アプライ・プログラムには、管理スレッド、ワーカー・スレッド、シリアライゼーション・スレッドの 3 つのスレッドがあります。キャプチャー・プログラムとアプライ・プログラムのワーカー・スレッドおよびシリアライゼーション・スレッドは、それぞれ別のものです。
- プログラムが正しく機能しているかどうかを、メッセージを受け取ることによって認識できます。一般的には、ワーカー・スレッド、管理スレッド、およびブルーニング・スレッドは作業状態にあります。スレッドが作業状態にある時には、実行しなければならぬタスクを実行しています。
- スレッドの状態
  - 作業中 (Is doing work)
    - 処理中です。
  - 存在します (Exists)
    - スレッドは存在するが、開始できない。IBM サービス・サポートにご連絡ください。
  - 開始済み (Was started)
    - スレッドは開始済みだが、初期化できない。スレッドが多すぎる、CPU が不足している、などのシステム・リソース上の問題がないか調べてください。
  - 初期化中 (Is initializing)
    - スレッドは初期化されるが、機能できない。IBM サービス・サポートにご連絡ください。
  - 休止中 (Is resting)
    - この状態は、キャプチャー・プログラムのスレッドだけに関係する。スレッドがこの状態にある時には、キャプチャー・プログラムが中断されており、操作の再開を待っている状態です。
  - 停止済み (Is stopped)
    - スレッドは実行されていません。スレッドが停止した理由を示すメッセージがないか、IBMSNAP APPLYTRAILまたはIBMSNAP CAPTRACE表を調べてください。たとえば、スレッドのブルーニングが停止したことを示すメッセージを受け取った場合は、その理由を確かめるために IBMSNAP CAPTRACE 表を調べてください。表が大きすぎて、すぐに整理したい場合は、キャプチャー・プログラムを停止し、ブルーニング・スレッドを開始するためにもう一度キャプチャー・プログラムを開始することができます。

## レプリケーション・アラート・モニター

## ■レプリケーション・アラート・モニター プログラム

- キャプチャー、アプライとは別のプログラム
- モニター・コントロール表を使用して、キャプチャーとアプライの処理をモニタリング
- 通知すべき状態が発生した場合は、自動的にアラートをEメールやポケベルに送信



## 解説:

- DB2 UDB V8 からできるようになった機能です。
- キャプチャー、アプライいずれかのプログラムが正常に作動していない場合には、それをいち早く検知する必要があります。自動化されたレプリケーション・アラート・モニターを使用して、レプリケーション環境の稼働およびパフォーマンスをチェックします。
- レプリケーション・アラート・モニターはDB2 for UNIX,DB2 for Windows,またはDB2 for z/OSで実行され、これらのプラットフォーム上で、また DB2 for iSeries で、データベース・サーバーをモニターできます。
- モニターした情報は、モニター・コントロール・サーバー上のモニター・コントロール表に保管されます。モニター・コントロール・サーバーはDB2 for UNIX,DB2 for Windows,またはDB2 for z/OSで使用できます。



## 設定と実行

### ■ 設定

- モニター・コントロール・サーバーの追加
- モニター・コントロール表の作成
- 連絡先情報の定義
- アラート条件の選択

### ■ 実行

- レプリケーション・センターからの開始
- コマンドでの開始

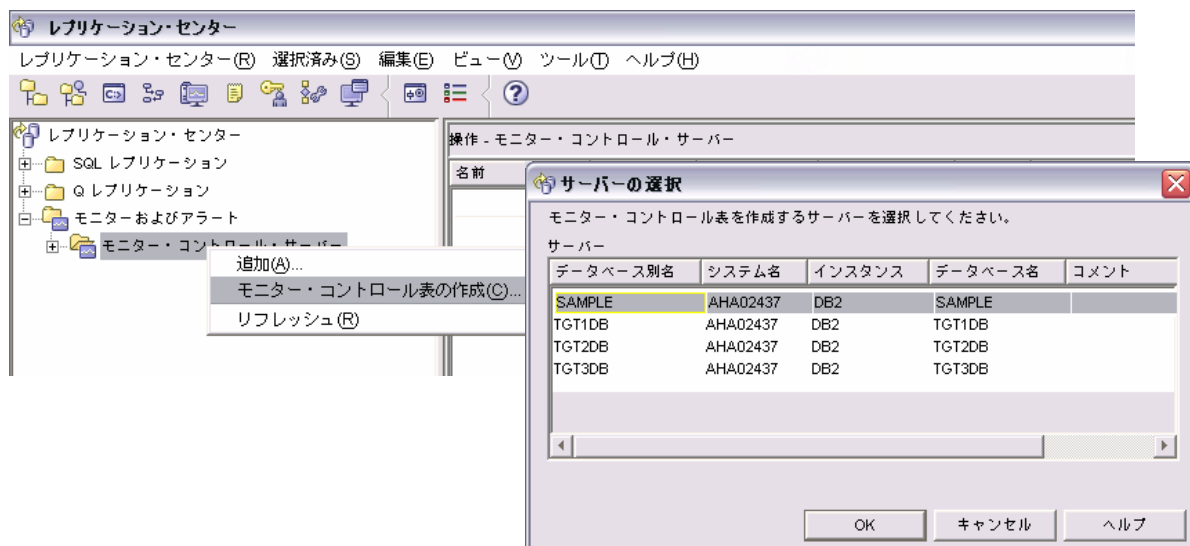


*blank page*



## コントロール表の作成

### ■ モニター・コントロール表を作成し、RCにモニター・コントロール・サーバーを追加



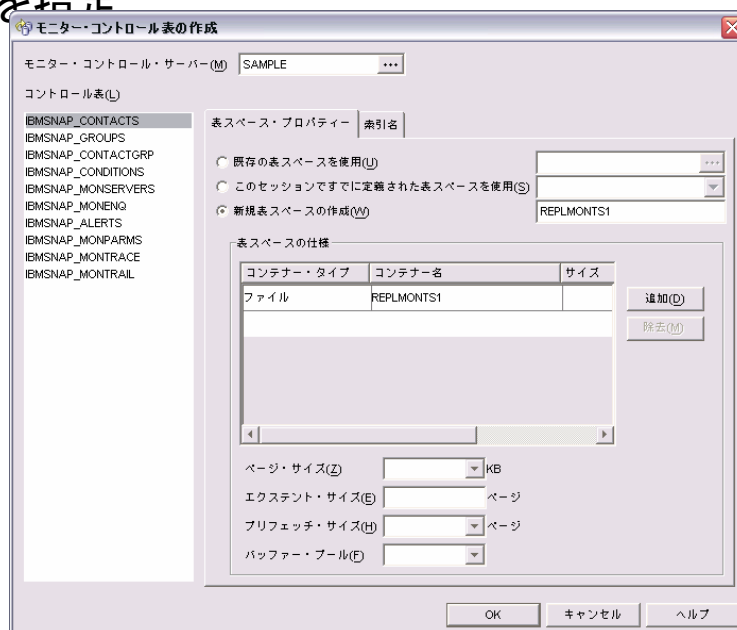
## 解説:

- モニター・プログラムは、キャプチャー・プログラム、キャプチャー・トリガー、およびアプライ・プログラムが使用するレプリケーション・コントロール表から分離されたコントロール表のセットを使用します。ユーザーは、これらのコントロール表をモニター・コントロール・サーバーで作成します。
- モニター・コントロール表にはスキーマの指定はできません。コントロール表は常にスキーマ ASN を使用するため、モニター・コントロール表を 1 セットだけサーバーに作成できます。UNIX, Windows および z/OS サーバーで作成することができます。
- 以下は、モニターに使用するコントロール表です。
  - ASN.IBMSNAP\_CONTACTGRP モニター・アラート連絡先表
    - 各グループに属する連絡先を保管します。
  - ASN.IBMSNAP\_ALERTS モニター・アラート表
    - 発行済みのアラートを保管します。
  - ASN.IBMSNAP\_CONDITIONS モニター条件表
    - キャプチャー・コントロール・サーバー、アプライ・コントロール・サーバーおよび現行のモニター・コントロール・サーバーでモニターするためのアラート条件が含まれます。アラート条件を満たした場合に通知する連絡先または連絡先のグループを、条件ごとにリストします。
  - ASN.IBMSNAP\_CONTACTS モニター連絡先表
    - この表には、名前、常用の E メール・アドレスおよび携帯電話の E メール・アドレスおよび各連絡先の記述が保管されます。アラート条件ごとの連絡先を、この表から選択します。
  - ASN.IBMSNAP\_GROUPS モニター・グループ表
    - この表には、連絡先の各グループの名前および記述が保管されます。アラート条件のセットの連絡先のグループを選択する場合、この表から選択します。
  - ASN.IBMSNAP\_MONENQ モニター・エンキュー表
    - この表を使用して、実行されているモニター処理がモニター修飾子ごとに 1 つだけであることを確認します。
  - ASN.IBMSNAP\_MONSERVERS モニター・サーバー表
    - この表は、キャプチャー・コントロール・サーバーおよびアプライ・コントロール・サーバーが、モニター修飾子で識別されたモニター・プログラムによって最後にモニターされた時刻を記録します。
  - ASN.IBMSNAP\_MONTRAIL モニター・トレール表
    - この表には、モニター・サイクルの履歴を保持します。
  - ASN.IBMSNAP\_MONTRACE モニター・トレース表
    - この表は、IBMSNAP\_CAPTRACE表およびIBMSNAP\_APPLYTRAIL表と同じように使用されます。モニター・プログラムのオペレーションのログです。



## コントロール表の作成

- キャプチャーやアプライと同様に、コントロール表を作成する表スペースを指定



## 解説:

- デフォルトの設定では、3つの表スペースを使用するようになっています。
- 表スペースと各コントロール表の関連は、以下のようになっています。

### REPLMONTS1

ASN.IBMSNAP\_CONTACTGRP  
ASN.IBMSNAP\_ALERTS  
ASN.IBMSNAP\_CONDITIONS  
ASN.IBMSNAP\_CONTACTS  
ASN.IBMSNAP\_GROUPS  
ASN.IBMSNAP\_MONENQ  
ASN.IBMSNAP\_MONSERVERS

### REPLMONTS2

ASN.IBMSNAP\_ALERTS

### REPLMONTS3

ASN.IBMSNAP\_MONTRAIL  
ASN.IBMSNAP\_MONTRACE



## コントロール表の作成

### ■ コントロール表作成用のSQLを生成し、実行



## 解説:

### ■ 生成されるSQL文

```
-- CONNECT TO SAMPLE USER XXXX USING XXXX;

CREATE TABLESPACE REPLMONT1
MANAGED BY DATABASE
USING
(
FILE 'REPLMONT1' 5M
);

CREATE TABLE ASN.IBMSNAP_CONTACTS(
CONTACT_NAME          VARCHAR(127) NOT NULL,
EMAIL_ADDRESS         VARCHAR(128) NOT NULL,
ADDRESS_TYPE          CHAR(1) NOT NULL,
DELEGATE              VARCHAR(127),
DELEGATE_START        DATE,
DELEGATE_END          DATE,
DESCRIPTION            VARCHAR(1024))
IN REPLMONT1;

CREATE UNIQUE INDEX ASN.IBMSNAP_CONTACTSX
ON ASN.IBMSNAP_CONTACTS(
CONTACT_NAME          ASC );

ALTER TABLE ASN.IBMSNAP_CONTACTS VOLATILE CARDINALITY;

CREATE TABLESPACE REPLMONT2
MANAGED BY DATABASE
USING
(
FILE 'REPLMONT2' 5M
);
```

```
CREATE TABLE ASN.IBMSNAP_ALERTS(
MONITOR_QUAL          CHAR(18) NOT NULL,
ALERT_TIME            TIMESTAMP NOT NULL,
COMPONENT             CHAR(1) NOT NULL,
SERVER_NAME           CHAR(18) NOT NULL,
SERVER_ALIAS          CHAR(8),
SCHEMA_OR_QUAL        VARCHAR(128) NOT NULL,
SET_NAME              CHAR(18) NOT NULL WITH DEFAULT ' ',
CONDITION_NAME        CHAR(18) NOT NULL,
OCCURRED_TIME         TIMESTAMP NOT NULL,
ALERT_COUNTER         SMALLINT NOT NULL,
ALERT_CODE            CHAR(10) NOT NULL,
RETURN_CODE           INT NOT NULL,
NOTIFICATION_SENT     CHAR(1) NOT NULL,
ALERT_MESSAGE         VARCHAR(1024) NOT NULL)
IN REPLMONT2;

CREATE INDEX ASN.IBMSNAP_ALERTX
ON ASN.IBMSNAP_ALERTS(
MONITOR_QUAL          ASC,
COMPONENT             ASC,
SERVER_NAME           ASC,
SCHEMA_OR_QUAL        ASC,
SET_NAME              ASC,
CONDITION_NAME        ASC,
ALERT_CODE            ASC);

ALTER TABLE ASN.IBMSNAP_ALERTS VOLATILE CARDINALITY;
```



## 解説:

## ■ つづき

```

CREATE TABLESPACE REPLMONTS3
MANAGED BY DATABASE
USING
(
FILE 'REPLMONTS3' 5M
);

CREATE TABLE ASN. IBMSNAP_MONPARMS (
MONITOR_QUAL          CHAR( 18) NOT NULL,
ALERT_PRUNE_LIMIT     INT WITH DEFAULT 10080,
AUTOPRUNE              CHAR( 1) WITH DEFAULT 'Y',
EMAIL_SERVER          VARCHAR(128),
LOGREUSE               CHAR( 1) WITH DEFAULT 'N',
LOGSTDOUT              CHAR( 1) WITH DEFAULT 'N',
NOTIF_PER_ALERT        INT WITH DEFAULT 3,
NOTIF_MINUTES          INT WITH DEFAULT 60,
MONITOR_ERRORS         VARCHAR(128),
MONITOR_INTERVAL      INT WITH DEFAULT 300,
MONITOR_PATH           VARCHAR(1040),
RUNONCE                CHAR( 1) WITH DEFAULT 'N',
TERM                  CHAR( 1) WITH DEFAULT 'N',
TRACE_LIMIT            INT WITH DEFAULT 10080,
ARCH_LEVEL             CHAR( 4) WITH DEFAULT '0810')
IN REPLMONTS3;

CREATE UNIQUE INDEX ASN. IBMSNAP_MONPARMSX
ON ASN. IBMSNAP_MONPARMS (
MONITOR_QUAL          ASC);

ALTER TABLE ASN. IBMSNAP_MONPARMS VOLATILE CARDINALITY;

```

```

CREATE TABLE ASN. IBMSNAP_GROUPS (
GROUP_NAME             VARCHAR(127) NOT NULL,
DESCRIPTION            VARCHAR(1024))
IN REPLMONTS1;

CREATE UNIQUE INDEX ASN. IBMSNAP_GROUPSX
ON ASN. IBMSNAP_GROUPS (
GROUP_NAME            ASC);

ALTER TABLE ASN. IBMSNAP_GROUPS VOLATILE CARDINALITY;

CREATE TABLE ASN. IBMSNAP_CONTACTGRP (
GROUP_NAME             VARCHAR(127) NOT NULL,
CONTACT_NAME           VARCHAR(127) NOT NULL)
IN REPLMONTS1;

CREATE UNIQUE INDEX ASN. IBMSNAP_CONTACTGPX
ON ASN. IBMSNAP_CONTACTGRP (
GROUP_NAME            ASC,
CONTACT_NAME          ASC);

ALTER TABLE ASN. IBMSNAP_CONTACTGRP VOLATILE CARDINALITY;

```



DB2 Universal Database

## 解説:

## ■ つづき

```

CREATE TABLE ASN. IBMSNAP_CONDITIONS (
MONITOR_QUAL          CHAR(18) NOT NULL,
SERVER_NAME           CHAR(18) NOT NULL,
COMPONENT              CHAR( 1) NOT NULL,
SCHEMA_OR_QUAL        VARCHAR(128) NOT NULL,
SET_NAME              CHAR(18) NOT NULL WITH DEFAULT ' ',
SERVER_ALIAS           CHAR( 8),
ENABLED               CHAR( 1) NOT NULL,
CONDITION_NAME        CHAR(18) NOT NULL,
PARM_INT              INT,
PARM_CHAR             VARCHAR(128),
CONTACT_TYPE          CHAR( 1) NOT NULL,
CONTACT               VARCHAR(127) NOT NULL)
IN REPLMONTS1;

CREATE UNIQUE INDEX ASN. IBMSNAP_MONCONDX
ON ASN. IBMSNAP_CONDITIONS (
MONITOR_QUAL          ASC,
SERVER_NAME           ASC,
COMPONENT             ASC,
SCHEMA_OR_QUAL        ASC,
SET_NAME              ASC,
CONDITION_NAME        ASC);

ALTER TABLE ASN. IBMSNAP_CONDITIONS VOLATILE CARDINALITY;

```

```

CREATE TABLE ASN. IBMSNAP_MONSERVERS (
MONITOR_QUAL          CHAR(18) NOT NULL,
SERVER_NAME           CHAR(18) NOT NULL,
SERVER_ALIAS          CHAR( 8),
LAST_MONITOR_TIME     TIMESTAMP NOT NULL,
START_MONITOR_TIME    TIMESTAMP,
END_MONITOR_TIME       TIMESTAMP,
LASTRUN               TIMESTAMP NOT NULL,
LASTSUCCESS           TIMESTAMP,
STATUS                SMALLINT NOT NULL)
IN REPLMONTS1;

CREATE UNIQUE INDEX ASN. IBMSNAP_MONSERVERX
ON ASN. IBMSNAP_MONSERVERS (
MONITOR_QUAL          ASC,
SERVER_NAME           ASC);

ALTER TABLE ASN. IBMSNAP_MONSERVERS VOLATILE CARDINALITY;

CREATE TABLE ASN. IBMSNAP_MONENQ (
MONITOR_QUAL          CHAR( 18) NOT NULL)
IN REPLMONTS1;

CREATE TABLE ASN. IBMSNAP_MONTRACE (
MONITOR_QUAL          CHAR(18) NOT NULL,
TRACE_TIME            TIMESTAMP NOT NULL,
OPERATION              CHAR( 8) NOT NULL,
DESCRIPTION            VARCHAR(1024) NOT NULL)
IN REPLMONTS3;

```



DB2 Universal Database

## 解説:

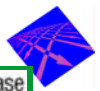
### ■ つづき

```
CREATE INDEX ASN. IBMSNAP_MONTRACEX
ON ASN. IBMSNAP_MONTRACE (
  MONITOR_QUAL          ASC,
  TRACE_TIME            ASC);

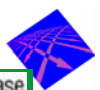
ALTER TABLE ASN. IBMSNAP_MONTRACE VOLATILE CARDINALITY;

CREATE TABLE ASN. IBMSNAP_MONTRAIL (
  MONITOR_QUAL          CHAR(18) NOT NULL,
  SERVER_NAME           CHAR(18) NOT NULL,
  SERVER_ALIAS          CHAR( 8),
  STATUS                SMALLINT NOT NULL,
  LASTRUN               TIMESTAMP NOT NULL,
  LASTSUCCESS           TIMESTAMP,
  ENDTIME               TIMESTAMP NOT NULL WITH DEFAULT,
  LAST_MONITOR_TIME     TIMESTAMP NOT NULL,
  START_MONITOR_TIME    TIMESTAMP,
  END_MONITOR_TIME      TIMESTAMP,
  SQLCODE               INT,
  SQLSTATE              CHAR(5),
  NUM_ALERTS            INT NOT NULL,
  NUM_NOTIFICATIONS     INT NOT NULL)
IN REPLMONTS3;

-- COMMIT;
```



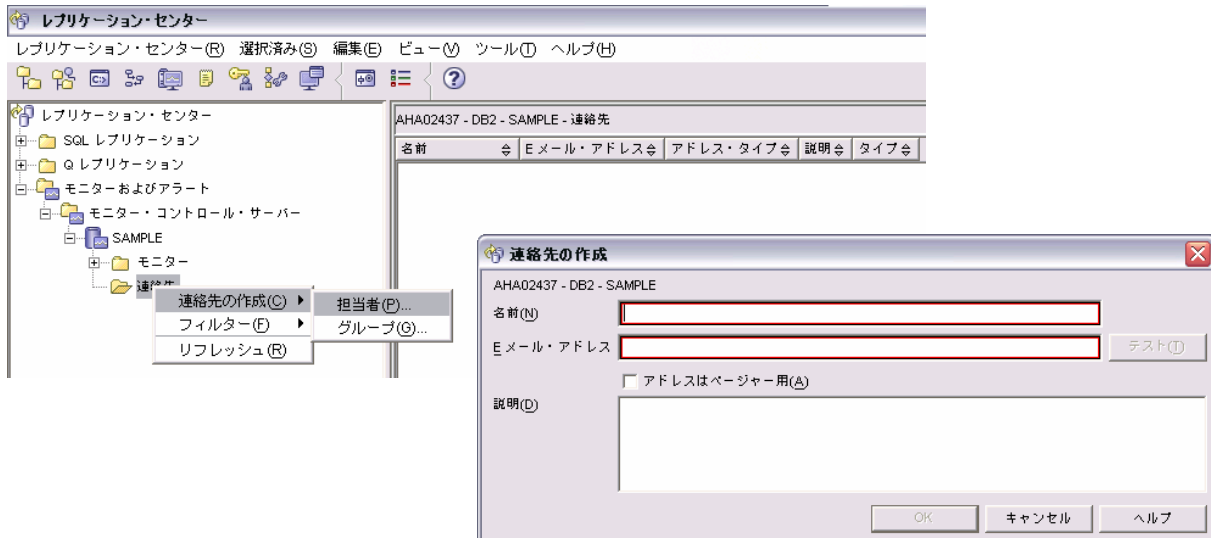
*blank page*



## 連絡先情報の定義

### ■ アラートを送信する連絡先を定義

- 通知する予定のある担当者、グループ情報を定義



## 解説:

### ■ 個人の連絡先

- 連絡先のリストに個人の連絡先を追加します。アラート条件を満たしたときにレプリケーション・アラート・モニターがメッセージを発行するイベントで、通知する個人の連絡先をこのリストから選択できます。
- 注: この連絡先のリストは、「タスク・センター」で使用する連絡先のリストとは別のものです。
- 連絡先を作成するとき、次の情報を入力します。
  - 連絡先の名前
  - 連絡先の E メール・アドレス (ページャーまたは通常の E メールを問わず)
  - 連絡先の記述
- 「テスト」ボタンをクリックして、指定された E メール・アドレスが正しいかどうかを確認する E メールが送信されます。

### ■ グループの連絡先

- レプリケーション・アラート・モニターにアラート・メッセージを送信させる個人の連絡先をグループ化します。たとえば、特定のキャプチャー・コントロール・サーバーにあるキャプチャー・プログラムについてのアラート・メッセージを 1 つのグループになった個人の連絡先に受信させ、そのサーバーに接続しているアプライ・コントロール・サーバーにあるコントロール表を伴うアプライ・プログラムについてアラート・メッセージを別のグループに受信させるとします。
- 注: この連絡先のリストは、「タスク・センター」で使用する連絡先のリストとは別のものです。
- 連絡先グループを作成するとき、次の情報を入力します。
  - グループの名前
  - グループの記述
  - グループを構成する個人の連絡先
- グループについて E メール・アドレスを用意する必要はありません。レプリケーション・アラート・モニターは、アラート・メッセージをグループ・メンバーの E メール・アドレスに送信します。

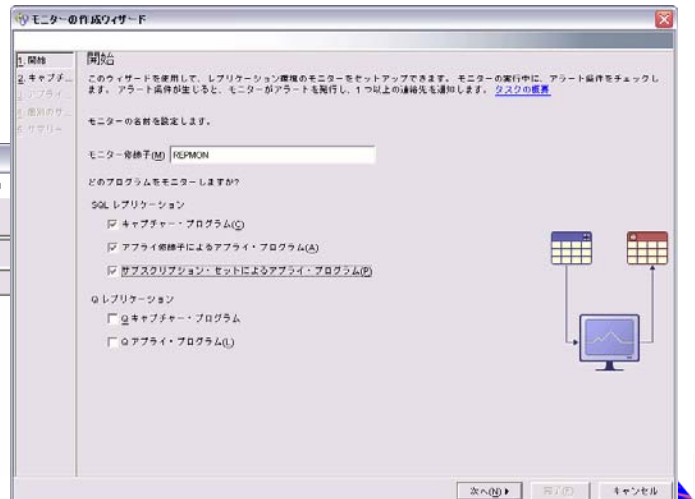
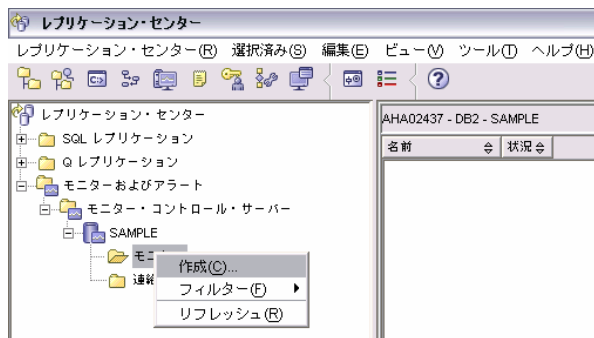
### ■ ここではIBMSNAP\_CONTACTS表へのINSERT文が生成されます。

- 例  
INSERT INTO ASN.IBMSNAP\_CONTACTS (CONTACT\_NAME,EMAIL\_ADDRESS,ADDRESS\_TYPE)  
VALUES ('Yumiko Matsui','yumiko@jp.ibm.com','E');

## モニタリング項目の選択

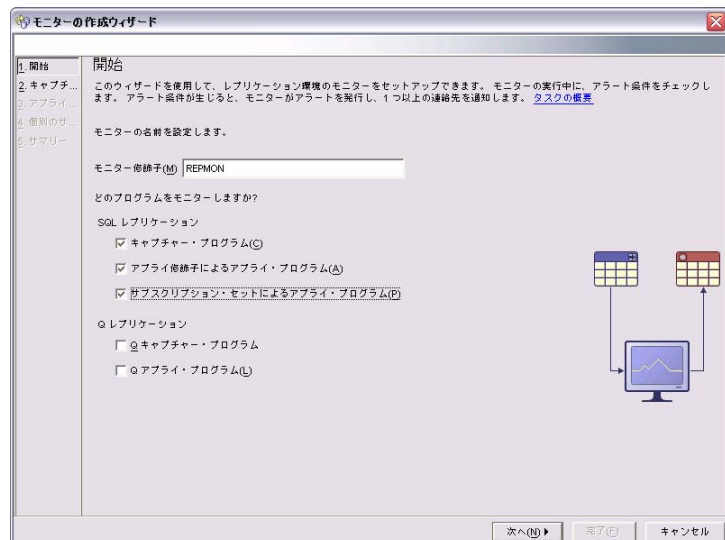
### ■ モニタすべきレプリケーションを選択

- SQLレプリケーション
  - キャプチャー・プログラム
  - アプライ修飾子によるアプライ・プログラム
  - サブスクリプションセットによるアプライ・プログラム
- Qレプリケーション
  - Qキャプチャー・プログラム
  - Qアプライ・プログラム



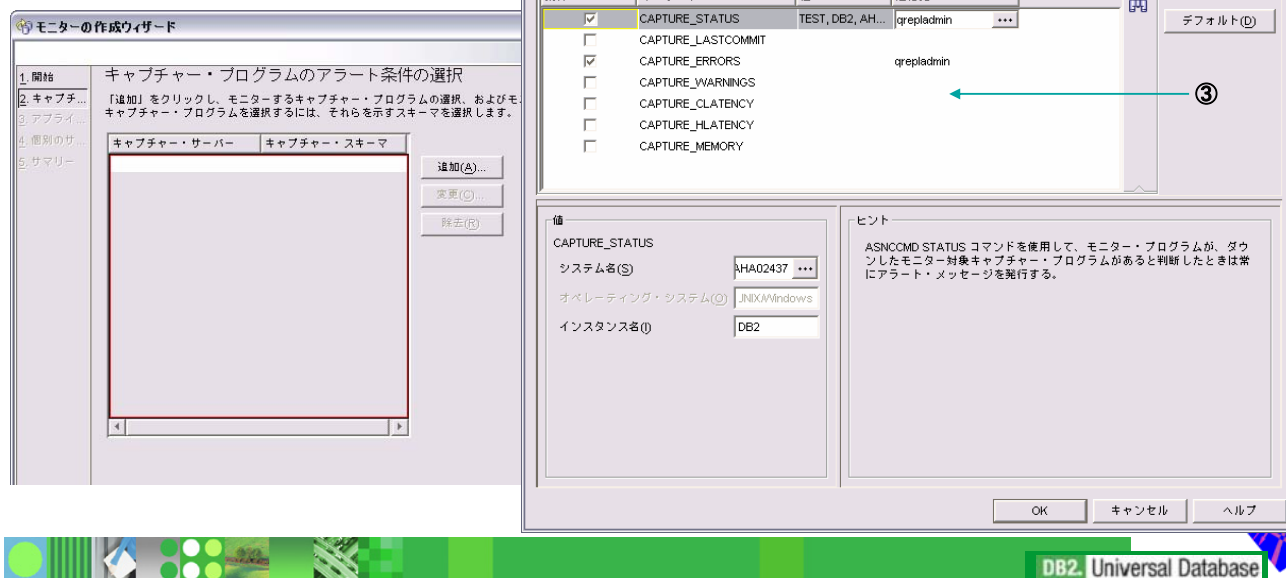
## 解説:

- 一つのモニター修飾子でSQLレプリケーションもQレプリケーションもモニタリングすることが可能です
  - SQLレプリケーション
    - キャプチャー・プログラム
    - アプライ修飾子によるアプライ・プログラム
    - サブスクリプションセットによるアプライ・プログラム
  - Qレプリケーション
    - Qキャプチャー・プログラム
    - Qアプライ・プログラム



## キャプチャー・プログラムのアラート条件の選択

- ①キャプチャー・コントロール・サーバー
- ②キャプチャー・スキーマ
- ③各アラート条件を指定

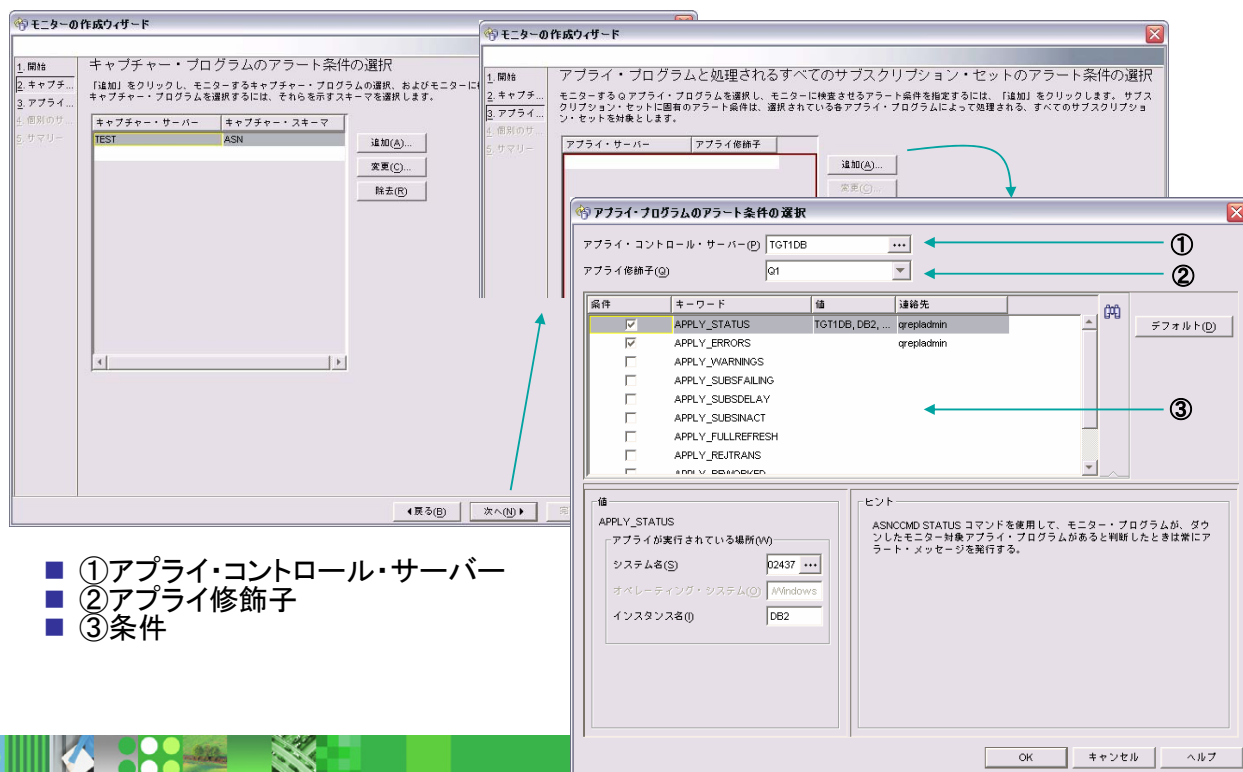


## 解説:

- 「キャプチャー・プログラムのアラート条件の選択」
  - 同じキャプチャー・コントロール・サーバーで異なるキャプチャー・スキーマで識別される1つまたは複数のキャプチャー・プログラムについて、アラート条件を選択してください。レプリケーション・アラート・モニターは、それらすべての条件に従って対象となるキャプチャー・プログラムを監視し、条件のいずれかを検出すると、ユーザーが指定した連絡先にメッセージを送信します。
- このウィンドウでOKをクリックすると、IBMSNAP\_CONDITIONSに情報を挿入するための以下のようなSQL文が生成されます。
  - 以下の例は、キャプチャーの状況ダウンとなんらかのエラーが発生した時にアラートを送信するためのものです。
    - 例
    - ```
INSERT INTO ASN.IBMSNAP_CONDITIONS
(MONITOR_QUAL, SERVER_NAME, COMPONENT, SCHEMA_OR_QUAL, SET_NAME, SERVER_ALIAS,
ENABLED, CONDITION_NAME, PARM_CHAR, CONTACT_TYPE, CONTACT)
VALUES
('MYREPMON', 'SAMPLE', 'C', 'ASN', ' ', 'SAMPLE', 'Y', 'CAPTURE_STATUS', ' ', 'C', 'Yumiko Matsui');
```
    - ```
INSERT INTO ASN.IBMSNAP_CONDITIONS
(MONITOR_QUAL, SERVER_NAME, COMPONENT, SCHEMA_OR_QUAL, SET_NAME, SERVER_ALIAS,
ENABLED, CONDITION_NAME, PARM_CHAR, CONTACT_TYPE, CONTACT)
VALUES
('MYREPMON', 'SAMPLE', 'C', 'ASN', ' ', 'SAMPLE', 'Y', 'CAPTURE_ERRORS', ' ', 'C', 'Yumiko Matsui');
```



## アプライ・プログラムのアラート条件の選択



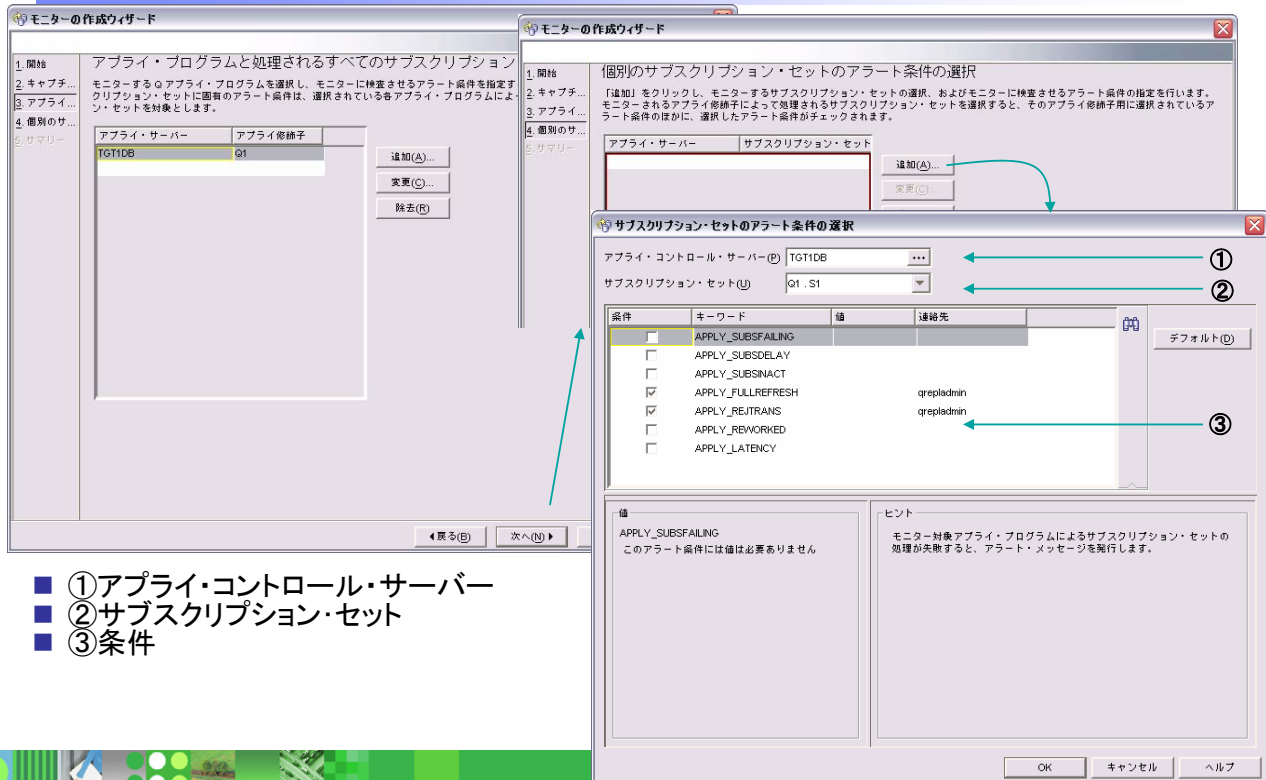
## 解説:

### ■ 「アプライ・プログラムのアラート条件の選択」

- 特定のサブスクリプション・セットおよびアプライ修飾子用にアラート条件を選択してください。レプリケーション・アラート・モニターは、選択されたすべての条件に従ってそれらのサブスクリプション・セットおよびアプライ修飾子を監視し、条件のいずれかを検出すると、ユーザーが指定した連絡先に通知します。



# サブスクリプション・セットのアラート条件の選択



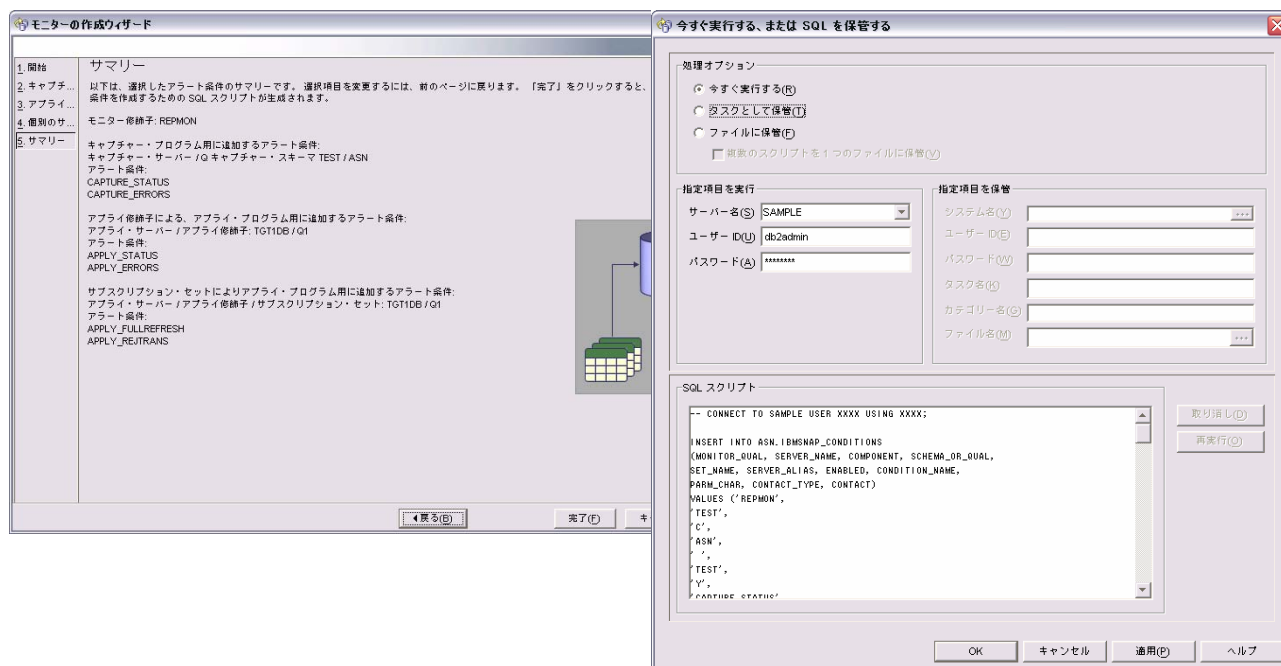
## 解説:

- アラート条件を付けたサブスクリプション・セットがあり、そのセットが含まれるアプライ修飾子についてアラート条件を選択すると、既存のサブスクリプション・セットが上書きされることはありませんが新しい条件が追加されます。
  - たとえば、SET1 がアプライ修飾子 APPLY1 によって処理されるとします。SET1 についてこのウィンドウをオープンし、アラート条件「APPLY\_REWORKED」(値 10 で) および「APPLY\_REJTRANS」を選択します。次に、APPLY1 についてアラート条件を選択します。このウィンドウをオープンし、アラート条件「APPLY\_REWORKED」(値 15 で) および「APPLY\_FULLREFRESH」を選択します。レプリケーション・アラート・モニターは、アラート条件「APPLY\_REWORKED」(値 10 で)、「APPLY\_REJTRANS」、および「APPLY\_FULLREFRESH」について SET1 を監視します。SET1 のアラート条件はいずれも、APPLY1 についてのアラート条件で上書きされませんが、SET1 に選択されたことのないアラート条件を APPLY1 から継承します。
- このウィンドウでOKをクリックすると、IBMSNAP\_CONDITIONSに情報を挿入するための以下のようなSQL文が生成されます。
  - 以下の例は、アプライのエラーとフルリフレッシュが発生した時にアラートを送信するためのものです。
    - 例
 

```
INSERT INTO ASN.IBMSNAP_CONDITIONS
(MONITOR_QUAL, SERVER_NAME, COMPONENT, SCHEMA_OR_QUAL, SET_NAME, SERVER_ALIAS, NABLED, CONDITION_NAME,
PARM_CHAR, CONTACT_TYPE, CONTACT)
VALUES
('MYREPMON','DPRDB','A','MYAPPLY','STAFFSET','DPRDB','Y','APPLY_ERRORS',' ','C','Yumiko Matsui');
```
    - ```
INSERT INTO ASN.IBMSNAP_CONDITIONS
(MONITOR_QUAL, SERVER_NAME, COMPONENT, SCHEMA_OR_QUAL, SET_NAME, SERVER_ALIAS, ENABLED, CONDITION_NAME,
PARM_CHAR, CONTACT_TYPE, CONTACT)
VALUES
('MYREPMON','DPRDB','A','MYAPPLY','STAFFSET','DPRDB','Y','APPLY_FULLREFRESH',' ','C','Yumiko Matsui');
```

## 解説:

### ■ サマリー



## 解説:

- 全ての項目を定義し終わると「サマリー」画面が表示されどのような条件を設定したかを確認できます。
- 「完了」ボタンを押すと、SQL文が生成され実行することができます。
- MONITOR\_INTERVALごとに、IBMSNAP\_MONTRAIL表を確認し、選択した条件に対するエラーがあればアラートを通知します。
- デフォルトのMONITOR\_INTERVALは5分なので、最悪アラートが5分遅れる場合があります。

## ASNCLPで定義

### ■レプリケーション・アラート・モニターをASNCLPを使用して作成

#### ● サンプルスクリプト

```
asnclp session set to sql replication;  
  
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;  
  
SET SERVER MONITOR TO DB SAMPLE;  
SET SERVER CAPTURE TO DB TEST;  
SET SERVER CONTROL TO DB TGTIDB;  
  
CREATE CONTROL TABLES FOR MONITOR CONTROL SERVER IN UW OTHERS USERSPACE1;  
  
CREATE CONTACT REPLADMIN EMAIL "ntakaya@jp.ibm.com" ;  
  
CREATE ALERT CONDITIONS FOR CAPTURE SCHEMA ASN MONITOR QUALIFIER MON NOTIFY CONTACT REPLADMIN (STATUS DOWN);  
  
CREATE ALERT CONDITIONS FOR APPLY QUALIFIER Q1 MONITOR QUALIFIER MON NOTIFY CONTACT REPLADMIN (STATUS DOWN);  
CREATE ALERT CONDITIONS FOR APPLY QUALIFIER Q1 SET NAME S1 MONITOR QUALIFIER MON NOTIFY CONTACT REPLADMIN (SUBSCRIPTIONS INACTIVE);
```

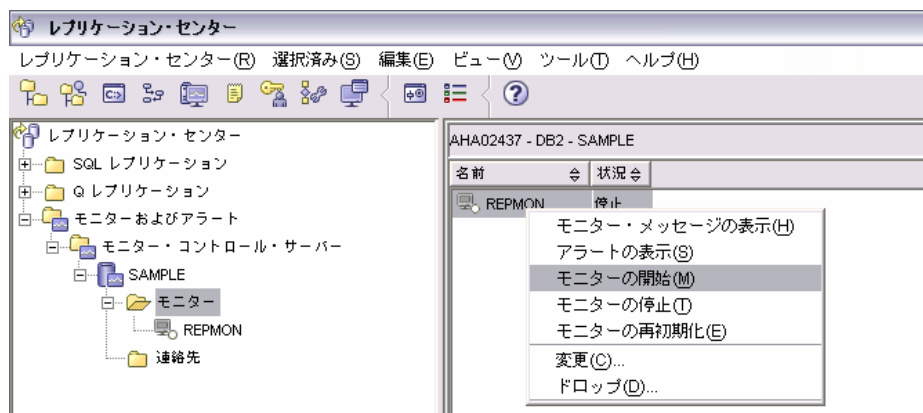


*blank page*



## レプリケーション・センターからの開始

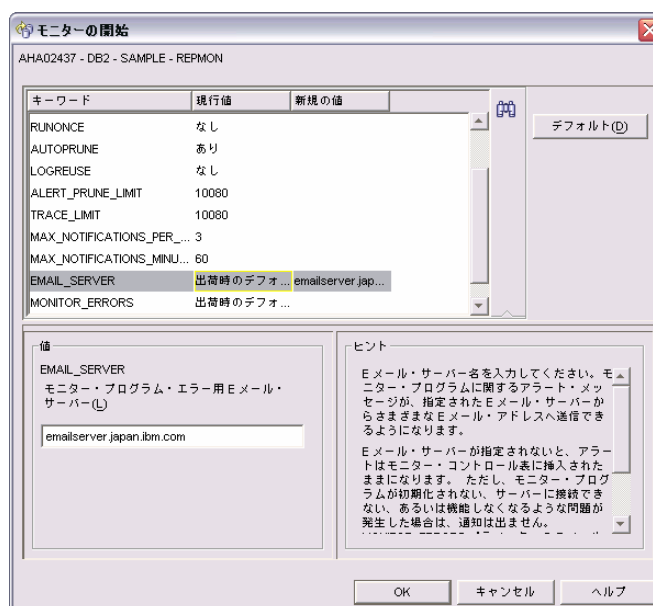
### ■ モニターを選択 右クリック ⇒ モニターの開始



## レプリケーション・センターからの開始

### ■ パラメータの指定

- 使用するメール・サーバーを指定



## 解説:

### ■ 「モニターの開始」

- モニター修飾子を開始します。モニター修飾子を連続的に実行するか、または 1 回のモニター循環だけにするかを選択できます。パラメーターを設定して、ログ・ファイルおよびパスワード・ファイルの書き込み場所を指定したり、レプリケーション・アラート・モニターが使用するコントロール表を管理したり、またはアラート・メッセージを管理することができます。最後に、モニター修飾子を実行中にエラーが発生したときに連絡する相手の電子メール・アドレスを入力できます。
- アラート送信に使用するメール・サーバーはここで指定してください。
  - 「タスク・センター」で使用するよう設定した、メール・サーバーがデフォルトになることはありません。
- モニタリングのインターバルはMONITOR\_INTERVALで変更することができます。(デフォルト5分)

### ■ このウィンドウでOKをクリックすると、以下のコマンドが生成されます。

- 例)
  - asnmon MONITOR\_SERVER=MONDB MONITOR\_QUAL=MYREPMON MONITOR\_PATH=d:\dprv8  
EMAIL\_SERVER=dbdcserv.fscjapan.ibm.com

### ■ モニター・プログラムをコマンドで開始する場合は、上記のコマンドを使用します。



  
*blank page*



## コマンドでの開始

## ■ 実行例

- `asnmon monitor_server=MONDB monitor_qual=MYMON`

## ■ Syntax

```

>>--asnmon-----monitor_qual=mon_qual-----
      |'-monitor_server=server-'|'-monitor_interval=n-'|'.-n-.'|'-runonce=++y++'|
>-----
>|'.-y-.'|'.-n-.'|'.-n-.'|'-alert_prune_limit=n-'|
      |'-autoprune=++n++'|'-logreuse=++y++'|'-logstdout=++y++'|
>-----
>|'-trace_limit=n-'|'-max_notifications_per_alert=n-'|'-max_notifications_minutes=n-'|
>-----
>|'.-asnpwd.aut.'|'-monitor_path=path-'|'.-.'|
      |'-pwdfile=++filepath++'|'-monitor_errors=--"--address--"'|
>-----><
>|'-email_server=servername-'|'.-n-.'|
      |'-console=++y++'|

```

**解説:**

- **monitor\_server=server**
  - レプリケーション・アラート・モニター・コントロール表が存在するモニター・コントロール・サーバーの名前
  - これを入力する場合には、このパラメーターを必ず最初のパラメーターにする
  - **Linux, UNIX, Windows:** モニター・コントロール・サーバーを指定しない場合、このパラメーターはデフォルトで DB2DBDFT 環境変数の値になる
  - **z/OS:** デフォルトは DSN です。
- **monitor\_qual=mon\_qual**
  - モニター修飾子
  - モニター修飾子は必ず指定する
  - モニター修飾子名には大/小文字の区別があり、最大 18 文字
- **monitor\_interval=n**
  - レプリケーション・アラート・モニター・プログラムを実行する頻度 (秒数) を指定
  - 省略値は 300 秒 (5 分)
  - **runonce** パラメーターを **y** に設定すると、レプリケーション・アラート・モニターはこのパラメーターを無視する
  - **重要:** この **monitor\_interval** パラメーターは、レプリケーション・アラート・モニター・プログラムにのみ影響を与える キャブチャ、およびアプライの各プログラムには影響を与えない
- **runonce=y/n**
  - レプリケーション・アラート・モニター・プログラムを 1 回だけ実行するかどうかを指定
  - **n (デフォルト)**
    - **monitor\_interval** パラメーターに指定された頻度で実行される
  - **y**
    - モニター・サイクルを 1 回だけ実行
    - **runonce** パラメーターを **y** に設定すると、**monitor\_interval** は無視される
- **autoprun=y/n**
  - IBMSNAP\_ALERTS 表内の行の自動整理を使用可能にするかどうかを指定
  - **y (デフォルト)**
    - **alert\_prune\_limit** パラメーターの値よりも古い、IBMSNAP\_ALERTS 表内の行を自動的に整理
  - **n**
    - 自動整理は使用不可



## 解説:

- **logreuse=y/n**
  - ログ・ファイル (db2instance.monitor\_server.mon\_qual.MON.log) を再利用するか、またはメッセージを付加するかを指定
  - n (デフォルト)
    - ログ・ファイルにメッセージを付加
  - y
    - ログ・ファイルを削除し、レプリケーション・アラート・モニター・プログラムの再始動時にそれを再作成し、ログ・ファイルを再利用
- **logstdout=y/n**
  - アラート・メッセージをどこに送信するかを指定
  - n (デフォルト)
    - ログ・ファイルにのみメッセージを送信
  - y
    - メッセージをログ・ファイルと標準出力 (stdout) の両方に送信
- **alert\_prune\_limit=n**
  - IBMSNAP\_ALERTS表に行を保持する期間 (分) を指定
  - この値よりも古い行はすべて整理される
  - 省略値は 10,080 分 (7 日)
- **trace\_limit=n**
  - IBMSNAP\_MONTRACE表内に何分保持されるとその行が整理の対象になるかを示す分数を指定
  - **trace\_limit** パラメーターの値よりも古いすべての IBMSNAP\_MONTRACE 行が、次の整理サイクルで整理される
  - 省略値は10,080 分 (7 日)
- **max\_notifications\_per\_alert=n**
  - このパラメーター値で指定された期間中にアラートが生じた場合に、同じアラートをユーザーに送信する最大回数を指定
  - このパラメーターは、同じアラートがユーザーに何回も再送されないようにするために使用する
  - 省略値は3
- **max\_notifications\_minutes=n**
  - **max\_notifications\_per\_alert** パラメーターと連動し、アラート条件が起きた期間を示す
  - 省略値は 60 分



## 解説:

- **pwdfile=filepath**
  - パスワード・ファイルの完全修飾名を指定
  - このファイルは asnpwd コマンドを使用して定義する
  - デフォルトのファイル名は、asnpwd.aut
- **monitor\_path=path**
  - ログ・ファイルのロケーションを指定
  - デフォルトは、asnmon コマンドが呼び出されたディレクトリー
- **monitor\_errors=address**
  - アラート・モニターがモニター・コントロール・サーバーに接続する前に致命的エラーが検出された場合、その通知を送信する宛先の E メール・アドレスを指定
  - このパラメーターを使用して、開始パラメーターが無効である、モニター修飾子が誤っている、データベースがダウンしている、またはその他のエラーのために、モニター・コントロール・サーバー接続が失敗したという通知を送信する
  - E メール・アドレスのテキストは二重引用符で囲む
  - 複数の E メール・アドレスを入力することもできる
  - E メール・アドレスはコンマで区切り、コンマの前後にスペースを入力してもそれらは無視される
- **email\_server=servername**
  - Eメールのサーバー・アドレスを指定
  - SMTP (Simple Mail Transfer Protocol) を指定して ASNMAIL 出力ルーチンを使用する場合にのみ 入力する
- **console=y/n**
  - アラート通知を z/OS コンソールに送信するかどうかを指定
  - このパラメーターを y (はい) に設定し、E メール・サーバーがすでに構成されている場合には、アラートが z/OS コンソールと E メール・サーバーの両方に送信される
  - n (デフォルト)
    - アラート通知を z/OS コンソールに送信しない
  - y
    - アラート通知を z/OS コンソールに送信する



## 送信されるアラート

### ■ 条件に合致した事象が発生すると、以下のようなアラートが送信される

#### ● E-Mailの場合

To: repladmin@company.com  
From: replmon@server.com  
Subject: Monitor: "REPMON" Alerts issued

ASN5129I MONITOR "REPMON". The Replication Alert Monitor on server "SAMPLE" reports an e-mail alert  
2005-10-13-14.46.09 1 ASN0552E Capture : "ASN" The program encountered an SQL error. The server  
name is "CORP". The SQL request is "PREPARE". The table name "PROD1.INVOICESCD".  
The SQLCODE is "-204". The SQLSTATE is "42704". The SQLERRMC is "PROD1.INVOICESCD".  
The SQLERRP is "readCD"  
2005-10-13-14.46.09 2 ASN5152W Monitor "MONQUAL". The current Capture latency exceeds the threshold  
value. The Capture control server is "CORP". The schema is "ASN". The Capture latency is "90"  
seconds. The threshold is "60" seconds  
2005-10-13-14.46.09 4 ASN5154W Monitor "MONQUAL". The memory used by the Capture program exceeds  
the threshold value. The capture control server is "CORP". The schema is "ASN".  
The amount of memory used is "34" megabytes. The threshold is "30" megabytes.



DB2 Universal Database

## 解説:

### ■ 日本語でも送信可能

差出人: takaya  
あて先: ntakaya@jp.ibm.com  
件名: ASN5134I MONITOR REPMON。アラートが発行されました。  
本文:  
"ASN5134I MONITOR REPMON : WorkerThread。アラートが発行されました。"  
"ASN5129I MONITOR REPMON : WorkerThread。サーバー SAMPLE 上のレプリケーション・アラート・モニターは、  
Eメール・アラートを報告しました。"  
"2005-10-13-14.46.09 1 ASN5150W MONITOR REPMON : WorkerThread。プログラム CAPTURE は実行されて  
いません。サーバーは TEST、スキーマは ASN です。"



DB2 Universal Database

## アラートの応用例

### ■ ASNMAILというEXITプログラムでユーザー独自の動作を実行することができる

- Windows環境で、例えばVisualBasicにてエラーメッセージをポップアップで表示させるようなプログラムをASNMAIL exit から呼び出すことも可能
- 作成したASNMAILのプログラムはSQLLIB¥BIN¥に保存する
  - サンプルプログラム ⇒ ~¥sqllib¥samples¥repl¥

#### 1. asnmail.bat 内に以下を記述

```
start asnmail_1.vbs %1 %2 %3 %4 %5 %6 %7 %8
```

#### 2. 呼び出す、asnmail\_1.vbs に以下を記述

```
for each item in WScript.Arguments
alert = alert & vbcrf & item
next
wscript.echo alert
```

#### 3. 二つのファイルを SQLLIB¥BIN¥ に保存する

```
C:¥SQLLIB¥BIN¥>dir | find "asnmail"
2005/09/06 18:33          170 asnmail.bat
2005/09/06 18:39          93 asnmail_1.vbs
```



## アラートの応用例

### 4. アラートモニターを実行する

```
> asnmon monitor_server=sample monitor_qual=MON email_server=takaya
```

email\_serverオプションを指定すると  
ASNMAIL exit ルーチンが呼び出される  
email\_server は実際に機能する必要はない

### 5. 以下のようなポップアップがアラートごとに表示される

