

# 目次

[概要](#)

[機能情報](#)

[方法論](#)

[データ分析](#)

[一般的な問題](#)

## 概要

ローカル UCCX データベースアクセスを必要とするコンタクトセンター Express 統一された (UCCX) アクティビティがどのようにゆっくり実行するかもしれませんがこの資料に記述されています。それにより UCCX データを問い合わせることができないために AppAdmin ページは労働力マネージャおよび他のパフォーマンスおよび安定性の問題影響を、ウォールボード クエリへの応答の遅延奪取するのに、長い時間をかける AppAdmin に、更新ゆっくりロードします。

CLI で入るコマンド `show process` コードは `uccxoninit` が多量の CPU を消費することを示します。 `uccxoninit` プロセスは UCCX サーバで動作する UCCX Informix データベース 例を表します。

## 機能情報

UCCX アプリケーションをサポートするデータベース エンジン は IBM の Informix です。UCCX の AppAdmin ページに追加され、UCCX アプリケーションによって生成される設定および履歴情報は UCCX Informix 例で保存されます。

UCCX アプリケーションはウォールボード アプリケーション、品質管理、人事管理およびカスタム史的記事の目的で情報を得るために UCCX データベースに直接アクセスするのに使用できる 3 人のユーザを提供します。

各ユーザのユーザ情報、各ユーザの権限および意図されていた目的はここに記述されています:

- **uccxhruser** - このユーザは UCCX データベースで多くの設定および履歴テーブルに選択権限があり、カスタム史的記事および Cisco Unified 人事管理 (WFM) のためにだけ使用する必要があります。このユーザが実行するクエリおよび保存された手順は複雑な、長期クエリを行うかもしれません。ウォールボード アプリケーションのために発生するように WFM ユーザ、これらのクエリおよび保存された手順典型的な史的記事のプロファイルが原因で頻繁に実行されるべきではないです。

多くのウォールボード アプリケーションが `uccxhruser` にアクセスできる履歴テーブルおよび設定の内で含まれているデータを必要とするがこのユーザを複雑実行するのに使用することを、ウォールボード アプリケーションの目的で UCCX データベースに対して技術的に度々行きますクエリにサポートしません。

- **uccxworkforce** - `uccxworkforce` ユーザはチーム、リソースおよびスーパーバイザ表にアクセスでき、Cisco Unified 品質管理 (QM) に使用する必要があります。人事管理は

uccxworkforce ユーザによってアクセスが不可能である履歴データ 表にアクセスを必要とすると同時に uccxhruser を使用する必要があります。

- **uccxwallboard** -このユーザは UCCX エンジンのメモリから書かれているリアルタイム統計情報のスナップショットが含まれているリアルタイム データベーステーブルのだけ選択権限があります。表 RTCSQsSummary および RTICDStatistics に制限される選択権限は uccxwallboard ユーザがウォールボード アプリケーションによってソースをたどられるように意図されている簡単で、非複雑なクエリを用いる UCCX データベースを頻繁に問い合わせるのに使用する必要があることを意味します。

## 方法論

UCCX リリース 10.0 と それ以降では、UCCX データベースのパフォーマンストレースを始めるために **utils uccx データベース dbperf 開始する <interval> <totalHours>** コマンドを入力して下さい。このコマンドの **interval** 引数はトレース 収集の周期性を判別し、**totalHours** 引数は無効である前にトレースが実行する時間の総量を判別します。これらのパラメータはオプションです。コマンドがデフォルト値 20 分のおよび実行されるとき彼らが規定されなければ 10 時間は使用されません。

たとえば、**utils** を入力して下さいデータベースのパフォーマンストレースを有効にし、30 分毎にパフォーマンス統計情報のデータを 24 時間収集するために **uccx データベース dbperf 開始する 30 24** が命じる。

CLI コマンドによって得られるデータを収集する指示はコマンド 出力で印刷されます。

```
admin:utils uccx database dbperf stop
Execution of dbperf has been stopped
Command Successful
admin:utils uccx database dbperf start 24 30
The script runs every 30 minutes over a total duration of 24 hours.
Please collect files after 24 hours

Use "file get activelog uccx/cli/dbperf_171013134928.log" to get the file
Use "file view activelog uccx/cli/dbperf_171013134928.log" to view the file
Command Successful
admin: █
```

与えられる **totalHours** の後でデータ収集は自動的に停止します。手動で データ収集を停止するために、**utils uccx データベース dbperf 停止**コマンドを入力して下さい。

```
admin:utils uccx database dbperf stop
Execution of dbperf has been stopped
Command Successful
admin: █
```

UCCX バージョンがリリース 9.0(2) またはそれ以前であり、**utils uccx データベース dbperf** コマンドが利用できなかったら、詳細事項に関してはテクニカル アシスタンス センタ (TAC) に連絡して下さい。

TAC はリモート サポート アカウント アクセスと Cisco バグ ID [CSCuc68413](#) に手動で接続された dbperf.sh スクリプトを実行します。

スクリプト実行をいつ手動で開始するかまたは判別するかとき CLI コマンドによって、周律および合計時間は、**uccxoninit** プロセスによって消費される CPU をかなり変動するか、または原因解析のための必要な情報を収集するために残りますそれらの期間の間に高く確認します。

dbperf トレース スクリプトによって集められるログを関連させるために CPU が変動するときさらに、定期的に判別する **show process load** コマンドを入力して下さい。

## データ分析

dbperf スクリプトの **onstat** の実行によって集められるログ- **g SES 0** は UCCX データベースに対して発行されるアクティブなクエリを示します。 **uccxoninit** プロセスの高 CPU は一般的に実行するのに長い時間をかける複雑なクエリの結果です。目標はほとんどのリソースを消費し、それらのクエリのためのソース クライアントを判別し、即時解決のためのクライアントからのクエリをディセーブルにし、常置解決のための長期クエリを最適化するクエリを判別することです。

dbperf スクリプトによって集められるログでは CPU の可能性が高い原因高い変動が **uccxoninit** による支えられた高 CPU 消費が処理するクエリを探して下さい。

クエリを疑って下さい:

- **uccxhruser** として接続されるセッションから発行されます-先に記述されていて、**uccxhruser** に設定およびヒストリテーブルの膨大な数から情報を選択する特権があるように。その結果、複合体は、複数のテーブルを渡る長期クエリ組み立て、UCCX データベースのパフォーマンス影響を持つ場合があります。絶対が、**uccxwallboard** におよび **uccxworkforce** ユーザに UCCX データベース内の表にそのような制限されたアクセスが、これらのユーザが発行するパフォーマンス影響をまずない引き起こす複雑なクエリあります。さらに、**uccxhrc** によって発行されるクエリは UCCX 史的記事クライアント (HRC) または UCCX データベースに対する Cisco Unified Intelligence Center (CUIC) によって発行されます。これらのクエリは静的で、関連した indices と共に修正され、クエリ パフォーマンスへの最小の影響用の書かれ、テストされ、調整することができません。
- 行って下さいヒストリテーブルの集中的なクエリを- UCCX データベースが実行するように要求するクエリは UCCX データベースに表を渡って倍数、選定された相当数の情報加入するかまたは非指標付けされたフィールドをパフォーマンス影響を引き起こす可能性があります操作します。

**uccxhruser** として動作する 1 HR 表を含む複雑なクエリを用いる例はここに示されています:

上述の例は上のクエリに返される結果があった前に頻繁に入るか、または定期的に入ったら UCCX データベースのパフォーマンス影響を引き起こす可能性があるホスト **WBBOX** からソースをたどられる **uccxhruser** によって入る複雑なクエリを示します。

まれが、UCCX データベースのパフォーマンスはまた組み込みパージ プロセスの結果として (**uccxoninit** プロセスの CPU 稼働率は高く変動するか、または残ります)、低下でき。パージ プロセスはデータベースのサイズを維持するために設定からデータおよび UCCX データベース内のヒストリテーブルを削除するように設計されています。パージはデータベースの内で含まれているデータベースまたは最も古いレコードのサイズに基づいていましたスケジュールすることができます。

パージ プロセスが動作するとき、データは 1 つのクエリと取除かれます。取除くことをレコードの量に繰り返して基づいてされません。これはパージが取除く必要がある多量のデータを検出

するこのデータを取除くために一つのクエリを発行することを意味します。

UCCX AppAdminページからのページ スケジュールまたはパラメータの修正により多量のデータを取除くためにページをスケジュールするために完了するのに時間をかけますこの単一クエリは、次にスケジュールされたページに場合があります。従って、それはデータベースのインスタンスの CPU稼働率の上で駆動します。

dbperf スクリプトの出力では、ページ クエリは見られる場合があります。それは `sp_purge` 保存された手順を呼出すユーザ `uccxuser` によって入る唯一のクエリであるはずで

## 一般的な問題

最近の Cisco TAC および Cisco 開発工学エクスペリエンスに基づいて、これらは `uccxoninit` プロセスの CPU使用率が高い状態を引き起こす最もよく見られる問題です:

- 企業のクライアントは `uccxhruser` として接続し、ヒストリテーブルとウォールボードカスタムレポート ソリューションを提供するために加入されるウォールボード表の頻繁な複雑なクエリを ( `RTICDStatistics` および `RTCSQsSummary` ) 実行します。ウォールボード使用に関しては、リアルタイム テーブルにだけ `uccxwallboard` ユーザおよび制限クエリを使用して下さい。ウォールボードからのまたはウォールボードと同じような周波数の歴史的なコンフィギュレーションテーブルを問い合わせる機能はサポートされません。
- クライアントはセカンダリ ノードの代わりにアクティブなマスター ノードのカスタム史的記事を提出するように試みます。非エンジン マスター ノードの史的記事を生成する保存された手順だけを、カスタムかデフォルト実行して下さい。CUIC および HRC は非マスター ノードのクエリをデフォルトで実行しますが、ときカスタム史的記事を開発して、開発者にこれらのクエリを実行するか、またはこれらの保存された手順を実行するノード選択がある。
- Cisco 人事管理 ( WFM ) は `startdatetime` フィールドでフィルタリングするように試みるために `ContactRoutingDetail` 表の複雑なクエリを発行します。インデックスはこの表のこのフィールドでデフォルトで作成されません、従ってこのクエリのパフォーマンスは粗末です。WFM 問題定期的にこのクエリ UCCX から WFM にデータを同期するため。この問題は Cisco バグ ID [CSCtz23710](#) でキャプチャされ、WFM リリース 9.0(1)SR4 で解決されます。この問題に直面する顧客は Cisco バグ ID [CSCtz23710](#) のための修正が含まれている WFM のバージョンにアップグレードする必要があります。
- パージしきい値は次にスケジュールされたページが多量のデータを取除くように試みることでそのような物修正されます。よりもむしろかなり単一 アップデートのページ パラメータを、ページ スケジュール修正繰り返して作られますページ コンフィギュレーションの変更間の数日と、修正して下さい。これはページ プロセスが削除オペレーションのパフォーマンスを改善する各パスのより小さいデータセットを取除くようにします。
- `DialingList` 表は膨大です。 `DialingList` 表は送信キャンペーンにアップロードされるすべての連絡先を保存します。UCCX リリース 8.0 および 8.5 では、数百万のレコードが送信キャンペーンにアップロードされた後、パフォーマンス上の問題は表それから ( `uccxoninit` プロセスおよび遅い AppAdmin ページ ロードの高CPU を引き起こす ) 問い合わせられます生じます。パフォーマンス上の問題を軽減するために、 `DialingList` 表をクリーンアップする cron ジョブ スクリプトのインストールのための TAC ケースをオープンして下さい。UCCX リリース 9.0 ではパフォーマンスを改善するために、インデックスは AppAdmin からより有効なクエリのためのこの表に追加されました。この変更はすべてほとんどの極端なケースの問題を解決しました。この問題に広範囲の修正を提供する歴史的連絡先のアクティブな連絡先のた

めの UCCX リリース 10.0 DialingList は 2 つの表に、1 および別のものでは分割されました。