Borland
A MICRO FOCUS COMPANY

Silk Central 15.0

报告数据集市

Micro Focus
575 Anton Blvd., Suite 510
Costa Mesa, CA 92626

2013-12-02

内容

概述

Silk Central 报告数据集市使得出于报告目的访问数据更加轻松。它可将数据从生产表移至用于创建高级报告的专用视图。优势包括：

- 清晰的表和视图命名，允许您迅速找到所需数据。
- 经过预处理的数据，让您可以访问聚合数据，而无需自行计算。
- 性能提升，因为报告可以使用更简单、快速的 SQL 查询。
- 更少的生产数据库负载依赖关系，这也能提高性能，消除负载峰值。

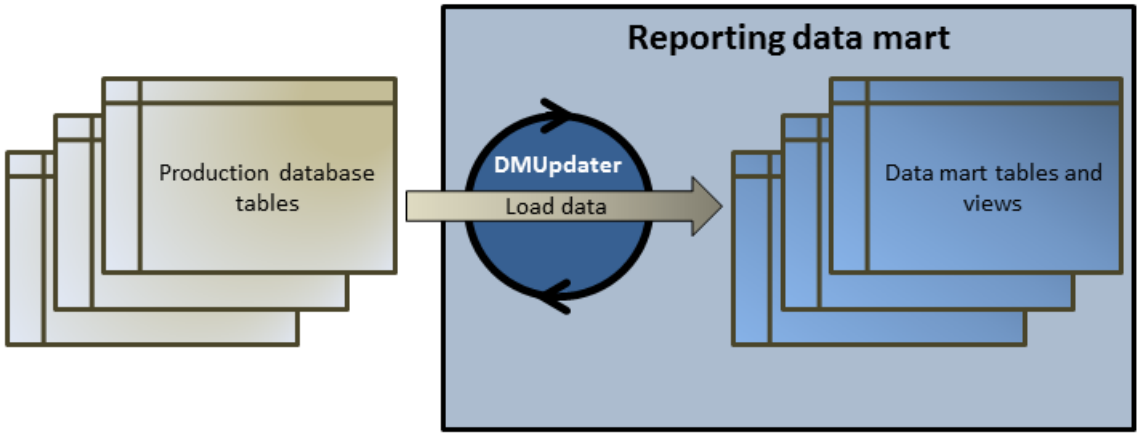
当前版本的数据集市已覆盖结果区域。未来版本的数据集市中将添加更多报告区域。当前提供的表和视图如下：

- `DM_TestStatus` 表，这是状态相关视图的基础。
- `RV_TestStatusExtended` 视图，提供特定测试执行的详细信息。
- `RV_LatestTestStatus` 视图，在执行计划和特定内部版本的上下文中，提供一个测试的最新测试运行的状态和扩展信息。
- `RV_MaxTestRunID` 视图，该帮助工具用于检索各测试、执行计划和内部版本组合的最新测试运行 ID。
- `RV_TestingCycleStatus` 视图，提供测试周期的状态信息。
- `RV_ExecutionPlanStatusPerBuild` 视图，在内部版本的上下文中，检索各执行计划的最新测试状态汇总。
- `RV_ExecutionPlanStatusRollup` 视图，在内部版本的上下文中，检索各执行计划或文件夹的通过、失败和未执行测试汇总。
- `RV_ConfigurationSuiteStatus` 视图，列出所有配置套件和各内部版本配置的状态计数。

体系结构

定期在后台提取生产数据库表中的数据，并将其加载到数据集市表和视图中，以简化和加速查询。只要数据库负载不过高，通常即可在提交更改后的一分钟之内得到此数据。如果您作为系统管理员登录，可以导航到 `http://<server>:<port>/sctm/check/db`，检查 **DM_TestStatus 表**，以查看数据加载过程的当前状态。

注: 如果您从不包含数据集市的 Silk Central 版本（13.0 之前的版本）更新，数据集市表和视图将使用生产系统的数据进行初始填充。此过程可能需要一些时间，具体取决于数据库的大小。此过程完成后即可访问数据。



如何使用数据集市创建报告

以下示例演示了如何使用数据集市视图创建有用的报告。

编写数据集市查询

1. 在菜单中，单击**报告 > 详细信息视图**。
2. 在**报告**树中，选择您要新报告在其中显示的文件夹。
   这将确定在目录树中存储报告的位置。
3. 单击工具栏上的  。此时将打开**创建新报告**对话框。
4. 键入新报告的名称。
   这是在**报告**树中显示的名称。
5. 如果您要将此报告用于其他用户，请选中**与其他用户共享此报告**复选框。
6. 在**说明**字段中键入报告的说明。
7. 单击**高级查询**可打开**报告数据查询**字段。 插入先前编写的代码或在字段中直接编写新代码。
   插入占位符列表有助于您使用预定义的函数占位符编辑 SQL 查询。 有关详细信息，请参阅*自定义报告的 SQL 函数*。

   注: 如果您手动编辑查询的 SQL 代码，请单击**检查 SQL** 以确认操作。
8. 单击**完成**以保存设置。

执行计划中的测试可靠性

问题

在持续集成环境中，最理想的测试方式是每天至少对每日内部版本执行一次测试，确保所测试应用程序的质量。为了解您的测试集在度量 AUT 质量方面的可靠程度，有时必须要查看结果随着时间推移而变化的情况。例如，您的测试集中可能存在状态频繁更改的测试，因此不具备真正的质量测量。

解决方案

使用数据集市视图 *RV_TestStatusExtended* 创建报告，列出特定执行计划上下文中特定测试的结果。这允许您查看该测试的结果随着时间推移而变化的情况。为方便起见，我们将结果列表缩小至与带标记的内部版本相关的结果，因而可以仅查看所测试应用程序的特定里程碑内部版本。此报告将在执行计划和内部版本的上下文中收集各个测试的测试结果数据。在以下查询中：

- 选择要从视图中显示的列。
- 按照希望调查的测试的 ID 以及该测试所属执行计划的 ID 缩小结果范围。
- 添加约束，以仅考虑带标记的内部版本。

```
SELECT TestName, ExecutionPlanName, VersionName, BuildName, TestRunID,
  PassedCount, FailedCount, NotExecutedCount
FROM RV_TestStatusExtended
WHERE TestID = ${TESTID|1|Test ID} AND ExecutionPlanID = ${EXECUTIONPLANID|1|
Execution Plan ID} AND BuildIsTagged = 1
ORDER BY BuildOrderNumber
```

SQL 查询的结果将得到选定执行计划内选定测试的所有测试运行。在以下示例中，可以看到对内部版本 `579_Drop2` 重新运行了测试：

TestName	Execution PlanName	Version Name	BuildName	TestRunID	Passed Count	FailedCount	NotExecutedCount
UI Tests	EN| SQL2012| IE9|IIS	3.0	579_Drop02	7741797	59	5	0
UI Tests	EN| SQL2012| IE9|IIS	3.0	579_Drop02	7745078	63	1	0
UI Tests	EN| SQL2012| IE9|IIS	3.0	593_Drop03	7787437	63	1	0
UI Tests	EN| SQL2012| IE9|IIS	3.0	605_Drop04	7848720	63	1	0

执行文件夹中的所有失败测试

问题

通常所有执行计划都组织在文件夹层次结构中，用于识别执行计划及其测试的不同相关区域或目的。执行计划在持续集成环境中定期触发，或在发布时间范围内不定时触发，以得到理想的执行统计信息 – 遗憾的是，这些信息仅限于每一个单独的执行计划。

有时，您需要获得为特定区域或目的而执行的所有测试的整体信息，以识别薄弱环节。

解决方案

使用数据集市视图 *RV_LatestTestStatus* 创建报告，返回特定执行计划层次级别的所有失败测试列表。

以下查询使用执行计划名称和内部版本名称等上下文信息，选择执行计划文件夹内的失败测试：

```
SELECT TestID, TestName, ExecutionPlanID, ExecutionPlanName, BuildName
FROM RV_LatestTestStatus lts
INNER JOIN TM_ExecTreePaths ON lts.ExecutionPlanID =
TM_ExecTreePaths.NodeID_pk_fk
WHERE TM_ExecTreePaths.ParentNodeID_pk_fk = ${executionFolderID|2179|
Execution Folder ID}
  AND StatusID IN (2, 9)
ORDER BY TestName
```

该查询执行以下操作：

- 使用视图 `RV_LatestTestStatus` 检索最新测试运行结果。
- 包含执行树层次结构 (`TM_ExecTreePaths`)，以查询该层次结构内所有执行计划的所有测试。
- 将应作为分析开始位置的顶层文件夹 ID 用作 `ParentNodeID_pk_fk`。
- 仅包含失败和未解决测试状态 (`StatusID IN (2,9)`)。

可以在 `StatusID` 表中查找 `TM_TestDefStatusNames` 数字 2 和 9。Silk Central 映射状态如下：

- 通过和不支持 -> 通过。
- 失败和未解决 -> 失败。
- 未执行和不适用 -> 未执行。

SQL 查询的结果将得到选定执行文件夹内上次运行失败的所有测试。

TestID	TestName	ExecutionPlanID	ExecutionPlanName	BuildName
14073	JUnitTestPackage	2184	CI Testing	352
14107	Volatile Tests	2191	Volatile Test	352

测试周期状态

问题

测试周期属于复杂的对象，因为其中包含有关手动测试工程师、测试、产品的不同内部版本和版本的信息，甚至还可能包含配置信息。为确保掌控测试周期，必须解答下面这些问题：

- 已经完成了多少测试？
- 每个内部版本通过或失败的测试有多少？
- 我的手动测试工程师是否已满负荷工作？他们能否再承担额外的工作？

解决方案

使用数据集市视图 *RV_TestingCycleStatus*，创建按测试工程师和内部版本显示测试周期状态的报告，这让您能获得有关按手动测试工程师、配置和内部版本分组的通过、失败、未执行测试数量的概述信息。

```
SELECT BuildName, TesterLogin, TesterExecutionName,
  PassedCount, FailedCount, NotExecutedCount
FROM RV_TestingCycleStatus
WHERE TestingCycleID = ${testingCycleID|3|Testing Cycle ID}
ORDER BY BuildOrderNumber, TesterLogin
```

该查询执行以下操作：

- 使用 `RV_TestingCycleStatus` 视图作为数据源，其中包含 `BuildName`、`TesterLogin` 和 `TesterExecutionName`，即为反映测试工程师、配置和测试而生成的名称。
- 将数据限制在您关注的测试周期 ID 范围内。

SQL 查询结果将显示各内部版本和测试工程师的状态。

BuildName	TesterLogin	TesterExecution Name	PassedCount	FailedCount	NotExecuted Count
352		No specific tester (Test Assets)	0	0	1
351	admin	admin (English| SQL2008|FF| Tomcat - Test Assets)	0	1	0
352	admin	admin (English| SQL2008|FF| Tomcat - Test Assets)	0	0	1
352	gmazzuchelli	gmazzuchelli (English| Oracle10g|IE8| Tomcat - Test Assets)	0	1	1
352	jallen	jallen (German| Oracle11g|FF|	1	1	0

BuildName	TesterLogin	TesterExecution Name	PassedCount	FailedCount	NotExecuted Count
		Tomcat - Test Assets)			
352	smiller	smiller (German|SQL2008|IE8|IIS - Test Assets)	1	1	0

对于未分配的测试，将创建一个“无特定测试工程师”组，并且以下几个值为空值：`TesterLogin`、`TesterFirstName` 和 `TesterLastName`。

如果您想根据手动测试工程师的绩效了解测试周期的情况，可对查询略作修改：

```
SELECT TesterLogin, TesterExecutionName, SUM(PassedCount) PassedCount,
  SUM(FailedCount) FailedCount, SUM(NotExecutedCount) NotExecutedCount
FROM RV_TestingCycleStatus
WHERE TestingCycleID = ${testingCycleID|3|Testing Cycle ID}
GROUP BY TesterLogin, TesterExecutionName
ORDER BY TesterLogin
```

查询进行了如下扩展：

- `GROUP BY TesterLogin, TesterExecutionName` 表示剩余列。
- `SUM()` 是用于聚合数字的计数器。

TesterLogin	TesterExecution Name	PassedCount	FailedCount	NotExecutedCount
	No specific tester (Test Assets)	0	0	1
admin	admin (English|SQL2008|FF|Tomcat - Test Assets)	0	1	1
gmazzuchelli	gmazzuchelli (English|Oracle10g|IE8|Tomcat - Test Assets)	0	1	1
jallen	jallen (German|Oracle11g|FF|Tomcat - Test Assets)	1	1	0
smiller	smiller (German|SQL2008|IE8|IIS - Test Assets)	1	1	0

执行树状态

问题

常见的做法是将执行计划置于表示不同测试区域或目的的层次结构中。在某些情况下，例如要了解测试状态，进而确定一个区域或目的的质量，您需要了解总体通过、失败和未执行计数。

解决方案

使用数据集市视图 *RV_ExecutionPlanStatusRollup* 为特定执行计划文件夹创建报告，以返回按内部版本分组的通过、失败和未执行计数。

```
SELECT BuildName, PassedCount, FailedCount, NotExecutedCount
FROM RV_ExecutionPlanStatusRollup
WHERE ExecutionFolderID = ${executionPlanID|43|Execution Plan ID}
```

该查询执行以下操作：

- 从 BuildName 视图选择 RV_ExecutionPlanStatusRollup 和状态计数。
- 指定您希望从中获取状态的顶层文件夹 (ExecutionFolderID)。

SQL 查询结果将显示选定文件夹所有执行计划内的测试运行状态，结果按内部版本聚合。

BuildName	PassedCount	FailedCount	NotExecutedCount
351	0	0	2
352	15	7	1

如果您想查看更多详细信息，例如选定层次结构内各执行计划的状态计数，可使用数据集市视图 *RV_ExecutionPlanStatusPerBuild*：

```
SELECT eps.BuildName, eps.ExecutionPlanID, SUM(eps.PassedCount) PassedCount,
  SUM(eps.FailedCount) FailedCount, SUM(eps.NotExecutedCount) NotExecutedCount
FROM RV_ExecutionPlanStatusPerBuild eps
INNER JOIN TM_ExecTreePaths etp ON eps.ExecutionPlanID = etp.NodeID_pk_fk
WHERE etp.ParentNodeID_pk_fk = ${execFolderID|44|Execution Folder ID}
GROUP BY eps.ExecutionPlanID, eps.BuildOrderNumber, eps.BuildName
ORDER BY eps.BuildOrderNumber, eps.ExecutionPlanID
```

该查询执行以下操作：

- 使用 RV_ExecutionPlanStatusPerBuild 视图访问特定于执行计划的数据（ExecutionPlanID 和 ExecutionPlanName）。先前使用的 RV_ExecutionPlanStatusRollup 视图包含预先聚合的数据（已汇总数据），不仅包含执行计划的结果，还包括文件夹节点的结果，因此不适于此目的。
- 对 JOIN 表执行 TM_ExecTreePath 操作，选定特定文件夹内的所有节点，以获取层次结构信息。
- 通过 ExecutionFolderID 指定顶层文件夹。TM_ExecutionTreePaths 表中还包含各执行计划的自引用，因此也可以使用 ParentNodeID_pk_fk 的执行计划 ID 运行此查询，以返回与特定执行计划对应的行。
- 添加 ORDER BY BuildOrderNumber 和 ExecutionPlanID 获取经过排序的结果，此时最早的内部版本及其执行计划将显示在最前。

SQL 查询结果将显示选定文件夹所有执行计划内的测试运行状态。

BuildName	ExecutionPlanID	PassedCount	FailedCount	NotExecutedCount
351	2307	0	0	2
352	2184	11	2	0
352	2185	0	3	0
352	2186	2	1	0
352	2187	1	0	0
352	2191	0	1	0
352	2307	1	0	1

配置套件状态

问题

配置套件允许您对多种配置（例如多种浏览器或操作系统）执行相同的一组测试。为了合理判断所测试应用程序的质量和可靠性，您需要跟踪每种配置的结果。

解决方案

使用数据集市视图 *RV_ConfigurationSuiteStatus* 创建为每个内部版本的各配置返回通过、失败和未执行计数的报告。

```
SELECT BuildName, ConfigurationName, PassedCount, FailedCount,
NotExecutedCount
FROM RV_ConfigurationSuiteStatus
WHERE ConfigurationSuiteID = ${configSuiteID|97|Configuration Suite ID}
ORDER BY BuildOrderNumber, ConfigurationName
```

该查询执行以下操作：

- 从 RV_ConfigurationSuiteStatus 视图检索测试运行的每个内部版本的状态计数。
- 将结果缩小到配置套件 (ConfigurationSuiteID)。

SQL 查询的结果显示每种配置的测试运行状态。

BuildName	ConfigurationName	PassedCount	FailedCount	NotExecutedCount
350	Chrome	0	1	0
350	Firefox	0	1	0
350	Internet Explorer	0	1	0
351	Chrome	1	0	0
351	Firefox	1	0	0
351	Internet Explorer	0	1	0
352	Chrome	1	0	0
352	Firefox	1	0	0
352	Internet Explorer	1	0	0

在本示例中，我们使用配置套件 ID 获取所有配置。另外也可以将结果限制为特定内部版本，此时必须在 where 子句中包含 BuildID、BuildName 或 BuildOrderNumber。

注: *RV_ConfigurationSuiteStatus* 视图仅包含聚合状态计数，不含任何特定于测试的数据。若要检索其他特定于测试的数据，可以使用 *RV_LatestTestStatus* 等视图。

故障排除

错误或缺失的数据

问题

在从数据集市表或视图中查询数据时，所列出的结果并非最新或有缺失。

解决方案

数据集市表和视图在后台定期更新，而非实时更新。因此，数据可能要等待几秒钟或者几分钟后才能载入数据集市表。如果您的系统负载较重，会对加载数据的后台进程的性能产生影响。原因在于其他进程的优先级更高，可能会临时阻止 DataMartUpdater 后台作业。请稍后再次运行查询，检索更新后的数据。

如果您作为系统管理员登录，可以导航到 `http://<server>:<port>/sctm/check/db`，检查 **DM_TestStatus 表**，以查看数据加载过程的当前状态。

注: 如果删除了一个测试，测试和依赖测试的测试运行都会从数据集市中删除。这同样适用于因清理测试包而删除的测试。

数据集市降低系统性能

问题

运行数据集市后，系统整体性能似乎有所降低，或者存在不一致的行为。

解决方案

尽管不应该采取这种做法，但您可以关闭数据集市，检查问题是否能得到解决：

1. 停止应用程序服务器服务。
2. 使用文本编辑器打开 `TmAppServerHomeConf.xml` 文件。此文件位于应用程序服务器 Silk Central 目录的 `/conf/appserver` 文件夹中。
3. 找到 `Config/DataMart/Enabled` XML 标记，将值设置为假。
4. 重新启动应用程序服务器服务。

参考：数据集市表和视图

以下数据集市表和视图可用于简化和加速报告。

DM_TestStatus

DM_TestStatus 表，这是状态相关视图的基础。

其他数据集市视图通常提供对详细信息数据的便捷访问，此表未提供测试名称等信息的直接访问。此表的键是 TestID、ExecutionPlanID、BuildID 和 TestRunID 列的组合。

行	说明
TestID	测试的标识符。
ExecutionPlanID	执行计划的标识符。
BuildID	内部版本的标识符。
TestRunID	测试运行的标识符。
ExecutionRunID	识别此结果将在哪个执行运行中生成。
StatusID	此测试运行的状态（请查看 TM_TestDefStatusNames）。
ReasonID	此测试运行的状态原因（请查看 TM_ResultStatusReasons）。可以为空。
PassedCount	所有通过的测试的总和，对于通用测试为 0 或 1，程序包测试根的此值可能更大。
FailedCount	所有失败的测试的总和，对于通用测试为 0 或 1，程序包测试根的此值可能更大。
NotExecutedCount	所有未执行的测试的总和，对于通用测试为 0 或 1，程序包测试根的此值可能更大。
ProjectID	此行所属项目的 ID。
TestStartTime	测试运行的开始时间 (UTC)。
ExecutionStartTime	执行运行的开始时间 (UTC)。
TestDurationInMilliseconds	测试运行的持续时间（毫秒）。
IsBlocked	表示测试运行已阻止/取消阻止的标记
DbChangedAt	报告数据集市最后更新此行的时间。

RV_TestStatusExtended 视图

RV_TestStatusExtended 视图，提供特定测试执行的详细信息。

此视图包含所有测试运行，而 *RV_LatestTestStatus* 视图仅包含一个执行计划、一个特定内部版本上下文中一个测试的最新测试运行。例如，您可以使用此视图创建*列出带标记内部版本的所有测试运行的报告*。此表的键是 TestID、ExecutionPlanID、BuildID 和 TestRunID 列的组合。

注: 如果删除了一个测试，测试和依赖测试的测试运行都会从数据集市中删除。这同样适用于因清理测试包而删除的测试。

行	说明
TestID	测试的标识符。
ExecutionPlanID	执行计划的标识符。
BuildID	内部版本的标识符。
TestRunID	测试运行的标识符。
ExecutionRunID	识别此结果将在哪个执行运行中生成。
StatusID	此测试运行的状态（请查看 TM_TestDefStatusNames）。
ReasonID	此测试运行的状态（请查看 TM_ResultStatusReasons）。可以为空。
PassedCount	所有通过的测试的总和，对于通用测试为 0 或 1，程序包测试根的此值可能更大。
FailedCount	所有失败的测试的总和，对于通用测试为 0 或 1，程序包测试根的此值可能更大。
NotExecutedCount	所有未执行的测试的总和，对于通用测试为 0 或 1，程序包测试根的此值可能更大。
ProjectID	此行所属项目的 ID。
TestStartTime	测试运行的开始时间 (UTC)。
ExecutionStartTime	执行运行的开始时间 (UTC)。
TestDurationInMilliseconds	测试运行的持续时间（毫秒）。
IsBlocked	表示测试运行已阻止/取消阻止的标记
DbChangedAt	报告数据集市最后更新此行的时间。
TestName	测试的名称。
TestDescription	测试的说明。
TestParentID	测试父项的 ID。
PlannedTimeInMinutes	此测试的计划时间（分钟）。
Reason	原因的名称。可以包含同时已删除的原因。
ExecutionPlanName	执行计划的名称。
ExecutionPlanDescription	执行计划的说明。
ExecutionParentFolderID	执行计划父项的 ID。
Priority	执行计划的优先级。0 = 低，1 = 中，2 = 高。
BuildName	用于此测试运行的内部版本的名称。
BuildDescription	内部版本的说明。
BuildOrderNumber	内部版本的序号。
BuildIsTagged	若内部版本带标记，则为 1，其他均为 0。
VersionID	内部版本所属版本的 ID。

行	说明
VersionName	版本名称。
VersionDescription	版本说明。
VersionOrderNumber	版本的序号。
ProductID	内部版本所属产品的 ID。
ProductCode	产品的名称。
ProductDescription	产品说明。
ProductOrderNumber	产品的序号。

RV_LatestTestStatus 视图

RV_LatestTestStatus 视图，在执行计划和特定内部版本的上下文中，提供一个测试的最新测试运行的状态和扩展信息。

使用 *RV_TestStatusExtended* 视图检索有关所有测试运行的信息。您可以使用此视图创建*列出执行文件夹内所有失败测试的报告*。此表的键是 TestID、ExecutionPlanID、BuildID 和 TestRunID 列的组合。

行	说明
TestID	测试的标识符。
ExecutionPlanID	执行计划的标识符。
BuildID	内部版本的标识符。
TestRunID	测试运行的标识符。
ExecutionRunID	识别此结果将在哪个执行运行中生成。
StatusID	此测试运行的状态（请查看 TM_TestDefStatusNames）。
ReasonID	此测试运行的状态（请查看 TM_ResultStatusReasons）。可以为空。
PassedCount	所有通过的测试的总和，对于通用测试为 0 或 1，程序包测试根的此值可能更大。
FailedCount	所有失败的测试的总和，对于通用测试为 0 或 1，程序包测试根的此值可能更大。
NotExecutedCount	所有未执行的测试的总和，对于通用测试为 0 或 1，程序包测试根的此值可能更大。
ProjectID	此行所属项目的 ID。
TestStartTime	测试运行的开始时间 (UTC)。
ExecutionStartTime	执行运行的开始时间 (UTC)。
TestDurationInMilliseconds	测试运行的持续时间（毫秒）。
IsBlocked	表示测试运行已阻止/取消阻止的标记
DbChangedAt	报告数据集市最后更新此行的时间。
TestName	测试的名称。

行	说明
TestDescription	测试的说明。
TestParentID	测试父项的 ID。
PlannedTimeInMinutes	此测试的计划时间（分钟）。
Reason	原因的名称。可以同时包含已删除的原因。
ExecutionPlanName	执行计划的名称。
ExecutionPlanDescription	执行计划的说明。
ExecutionParentFolderID	执行计划父项的 ID。
Priority	执行计划的优先级。0 = 低，1 = 中，2 = 高。
BuildName	用于此测试运行的内部版本的名称。
BuildDescription	内部版本的说明。
BuildOrderNumber	内部版本的序号。
BuildIsTagged	若内部版本带标记，则为 1，其他均为 0。
VersionID	内部版本所属版本的 ID。
VersionName	版本名称。
VersionDescription	版本说明。
VersionOrderNumber	版本的序号。
ProductID	内部版本所属产品的 ID。
ProductCode	产品的名称。
ProductDescription	产品说明。
ProductOrderNumber	产品的序号。

RV_MaxTestRunID 视图

RV_MaxTestRunID 视图，该帮助工具用于检索各测试、执行计划和内部版本组合的最新测试运行 ID。

此表的键是 TestID、ExecutionPlanID 和 BuildID 列的组合。

行	说明
TestID	测试的标识符。
ExecutionPlanID	执行计划的标识符。
BuildID	内部版本的标识符。
MaxTestRunID	标识在执行计划和内部版本上下文中测试的最新测试运行。

RV_TestingCycleStatus

RV_TestingCycleStatus 视图，提供测试周期的状态信息。

您可使用此视图创建*显示测试周期当前状态的报告*。

TestingCycleID 表示测试周期，TesterExecutionID（以及 TesterExecutionName、UserID、CapacityInCycle、TesterLogin、TesterFirstName 和 TesterLastName）用于识别测试周期中已分配的测试工程师。对于尚未分配到特定测试工程师的测试，UserID、CapacityInCycle、TesterLogin、TesterFirstName 和 TesterLastName 为 null。此表的键是 TesterExecutionID 和 BuildID 列的组合。

行	说明
TestingCycleID	测试周期的标识符。
TesterExecutionID	标识分配到特定测试工程师的测试组。
TesterExecutionName	为分配到特定测试工程师的测试组生成的名称。
UserID	测试工程师的用户 ID。
CapacityInCycleInMinutes	此用户在这个测试周期中的容量（分钟）。
TesterLogin	测试工程师的登录名称。
TesterFirstName	测试工程师的名字。
TesterLastName	测试工程师的姓氏。
PassedCount	所有已通过测试的汇总。
FailedCount	所有失败测试的汇总。
NotExecutedCount	所有未执行测试的汇总。
ProjectID	项目标识符。
BuildID	内部版本的标识符。
BuildName	用于此测试运行的内部版本的名称。
BuildDescription	内部版本的说明。
BuildOrderNumber	内部版本的序号。
BuildIsTagged	若内部版本带标记，则为 1，其他均为 0。
VersionID	内部版本所属版本的 ID。
VersionName	版本名称。
VersionDescription	版本说明。
VersionOrderNumber	版本的序号。
ProductID	内部版本所属产品的 ID。
ProductCode	产品的名称。
ProductDescription	产品说明。
ProductOrderNumber	产品的序号。

RV_ExecutionPlanStatusPerBuild

RV_ExecutionPlanStatusPerBuild 视图，在内部版本的上下文中，检索各执行计划的最新测试状态汇总。

不考虑文件夹和子节点。您可使用此视图创建*显示文件夹内各执行计划的测试运行状态的报告*。与 *RV_ExecutionPlanStatusRollup* 相比，此视图略有性能优势，因为在检索数据时不考虑层次结构。此表的键是 ExecutionPlanID 和 BuildID 列的组合。

行	说明
ExecutionPlanID	执行计划的标识符。
BuildID	内部版本的标识符。
ExecutionPlanName	执行计划的名称。
ExecutionParentFolderID	执行计划父项的 ID。
PassedCount	所有已通过测试的汇总。
FailedCount	所有失败测试的汇总。
NotExecutedCount	所有未执行测试的汇总。
ProjectID	执行计划所属项目的 ID。
BuildName	用于此测试运行的内部版本的名称。
BuildDescription	内部版本的说明。
BuildOrderNumber	内部版本的序号。
BuildIsTagged	若内部版本带标记，则为 1，其他均为 0。
VersionID	内部版本所属版本的 ID。
VersionName	版本名称。
VersionDescription	版本说明。
VersionOrderNumber	版本的序号。
ProductID	内部版本所属产品的 ID。
ProductCode	产品的名称。
ProductDescription	产品说明。
ProductOrderNumber	产品的序号。

RV_ExecutionPlanStatusRollup

RV_ExecutionPlanStatusRollup 视图，在内部版本的上下文中，检索各执行计划或文件夹的通过、失败和未执行测试汇总。

对于文件夹，计数器包含所有子文件夹的数字。您可使用此视图创建*显示文件夹内所有测试运行状态的报告*。此表的键是 ExecutionFolderID 和 BuildID 列的组合。

行	说明
ExecutionFolderID	执行计划的标识符。
BuildID	内部版本的标识符。
PassedCount	所有已通过测试的汇总。
FailedCount	所有失败测试的汇总。
NotExecutedCount	所有未执行测试的汇总。

行	说明
ProjectID	执行计划所属项目的 ID。
BuildName	用于此测试运行的内部版本的名称。
BuildDescription	内部版本的说明。
BuildOrderNumber	内部版本的序号。
BuildIsTagged	若内部版本带标记，则为 1，其他均为 0。
VersionID	内部版本所属版本的 ID。
VersionName	版本名称。
VersionDescription	版本说明。
VersionOrderNumber	版本的序号。
ProductID	内部版本所属产品的 ID。
ProductCode	产品的名称。
ProductDescription	产品说明。
ProductOrderNumber	产品的序号。

RV_ConfigurationSuiteStatus

RV_ConfigurationSuiteStatus 视图，列出所有配置套件和各内部版本配置的状态计数。

您可使用此视图创建*显示配置套件内各配置的所有测试运行的状态的报告*。此表的键是 ConfigurationID 和 BuildID 列的组合。

行	说明
ConfigurationSuiteID	配置套件的标识符。
ConfigurationID	配置的标识符。
ConfigurationName	配置名称。
BuildID	内部版本的标识符。
PassedCount	所有已通过测试的汇总。
FailedCount	所有失败测试的汇总。
NotExecutedCount	所有未执行测试的汇总。
ProjectID	此行所属项目的 ID。
BuildName	用于此测试运行的内部版本的名称。
BuildDescription	内部版本的说明。
BuildOrderNumber	内部版本的序号。
BuildIsTagged	若内部版本带标记，则为 1，其他均为 0。
VersionID	内部版本所属版本的 ID。
VersionName	版本名称。
VersionDescription	版本说明。

行	说明
VersionOrderNumber	版本的序号。
ProductID	内部版本所属产品的 ID。
ProductCode	产品的名称。
ProductDescription	产品说明。
ProductOrderNumber	产品的序号。