StoneWare Stored Procedure Documentation

The following documentation provides reference documentation for each Stored Procedure, including execution parameters and examples for use.

The following queries are covered:

Author: Cheron Fletcher
Date: 4/20/24

Description: Get all outstanding issues (all products)

Stored Procedure Name: `GetOutstanding`

Parameters:
none

Returns:
A List of Issues, or none if no matches are found.

Uses the following "Issue" Model:

```
    public class Issue
    {
        public required decimal TicketId { get; set; }
        public required string Product {  get; set; }
        public required decimal Version { get; set; }
        public required string OS { get; set; }
        public required string Status { get; set; }
        public required string Description { get; set; }
        public required DateTime StartDate { get; set; }
    }
```

Call the stored procedure in C# as follows:

```
            List<Issue> issues = new List<Issue>();
            string connectionstring =
"Server=(localdb)\\mssqllocaldb;Database=Stoneware;Trusted_Connection=True";
            using(var connection = new SqlConnection(connectionstring))
            {
                using(SqlCommand command = new SqlCommand("GetOutstanding", connection))
                {
                    connection.Open();
                    command.CommandType = CommandType.StoredProcedure;
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            issues.Add(new Issue
                            {
                                TicketId = (decimal)reader["IssueID"],
                                Product = (string)reader["ProductName"],
                                Version = (decimal)reader["VersionID"],
                                OS = (string)reader["OSName"],
                                Status = (string)reader["StatusType"],
                                Description = (string)reader["Description"],
                                StartDate = (DateTime)reader["StartDate"]
                            });
                        }
                    }
                    connection.Close();
                }
            }
```

Author: Cheron Fletcher
Date: 4/20/24

Description: Get all resolved issues (all products)

Stored Procedure Name: `GetResolved`

Parameters:
none

Returns:
A List of Issues, or none if no matches are found.

Uses the following "Issue" Model:

```
    public class Issue
    {
        public required decimal TicketId { get; set; }
        public required string Product {  get; set; }
        public required decimal Version { get; set; }
        public required string OS { get; set; }
        public required string Status { get; set; }
        public required string Description { get; set; }
        public required string Resolution { get; set; }
        public required DateTime StartDate { get; set; }
        public required DateTime EndDate { get; set; }
    }
```

Call the stored procedure in C# as follows:

```
            List<Issue> issues = new List<Issue>();
            string connectionstring =
"Server=(localdb)\\mssqllocaldb;Database=Stoneware;Trusted_Connection=True";
            using(var connection = new SqlConnection(connectionstring))
            {
                using(SqlCommand command = new SqlCommand("GetResolved", connection))
                {
                    connection.Open();
                    command.CommandType = CommandType.StoredProcedure;
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            issues.Add(new Issue
                            {
                                TicketId = (decimal)reader["IssueID"],
                                Product = (string)reader["ProductName"],
                                Version = (decimal)reader["VersionID"],
                                OS = (string)reader["OSName"],
                                Status = (string)reader["StatusType"],
                                Description = (string)reader["Description"],
                                Resolution = (string)reader["Resolution"],
                                StartDate = (DateTime)reader["StartDate"],
                                EndDate = (DateTime)reader["EndDate"]
                            });
                        }
                    }
                    connection.Close();
                }
            }
```

Author: Cheron Fletcher
Date: 4/20/24

Description: Get all outstanding issues for a product (all versions)

Stored Procedure Name: `GetOutstandingSingleProduct`

Parameters:
@ProductName (as string)

Returns:
A List of Issues, or none if no matches are found.

Uses the following "Issue" Model:

```
    public class Issue
    {
        public required decimal TicketId { get; set; }
        public required string Product {  get; set; }
        public required decimal Version { get; set; }
        public required string OS { get; set; }
        public required string Status { get; set; }
        public required string Description { get; set; }
        public required DateTime StartDate { get; set; }
    }
```

Call the stored procedure in C# as follows:

```
            List<Issue> issues = new List<Issue>();
            string connectionstring =
"Server=(localdb)\\mssqllocaldb;Database=Stoneware;Trusted_Connection=True";
            using(var connection = new SqlConnection(connectionstring))
            {
                using(SqlCommand command = new SqlCommand("GetOutstandingSingleProduct",
connection))
                {
                    connection.Open();
                    command.CommandType = CommandType.StoredProcedure;
                    //add Parameters
                    command.Parameters.AddWithValue("@ProductName", "Day Trader Wannabe");
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            issues.Add(new Issue
                            {
                                TicketId = (decimal)reader["IssueID"],
                                Product = (string)reader["ProductName"],
                                Version = (decimal)reader["VersionID"],
                                OS = (string)reader["OSName"],
                                Status = (string)reader["StatusType"],
                                Description = (string)reader["Description"],
                                StartDate = (DateTime)reader["StartDate"]
                            });
                        }
                    }
                    connection.Close();
                }
            }
```

Author: Cheron Fletcher
Date: 4/20/24

Description: Get all resolved issues for a product (all versions)

Stored Procedure Name: `GetResolvedSingleProduct`

Parameters:
@ProductName (as string)

Returns:
A List of Issues, or none if no matches are found.

Uses the following "Issue" Model:

```
    public class Issue
    {
        public required decimal TicketId { get; set; }
        public required string Product {  get; set; }
        public required decimal Version { get; set; }
        public required string OS { get; set; }
        public required string Status { get; set; }
        public required string Description { get; set; }
        public required string Resolution { get; set; }
        public required DateTime StartDate { get; set; }
        public required DateTime EndDate { get; set; }
    }
```

Call the stored procedure in C# as follows:

```
            List<Issue> issues = new List<Issue>();
            string connectionstring =
"Server=(localdb)\\mssqllocaldb;Database=Stoneware;Trusted_Connection=True";
            using(var connection = new SqlConnection(connectionstring))
            {
                using(SqlCommand command = new SqlCommand("GetResolvedSingleProduct",
connection))
                {
                    connection.Open();
                    command.CommandType = CommandType.StoredProcedure;
                    //add Parameters
                    command.Parameters.AddWithValue("@ProductName", "Day Trader Wannabe");
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            issues.Add(new Issue
                            {
                                TicketId = (decimal)reader["IssueID"],
                                Product = (string)reader["ProductName"],
                                Version = (decimal)reader["VersionID"],
                                OS = (string)reader["OSName"],
                                Status = (string)reader["StatusType"],
                                Description = (string)reader["Description"],
                                Resolution = (string)reader["Resolution"],
                                StartDate = (DateTime)reader["StartDate"],
                                EndDate = (DateTime)reader["EndDate"]
                            });
                        }
                    }
                    connection.Close();
                }
            }
```

Author: Cheron Fletcher
Date: 4/20/24

Description: Get all outstanding issues for a product (single version)

Stored Procedure Name: `GetOutstandingSingleProductSingleVersion`

Parameters:
@ProductName (as string)
@Version (as decimal)

Returns:
A List of Issues, or none if no matches are found.

Uses the following "Issue" Model:

```
    public class Issue
    {
        public required decimal TicketId { get; set; }
        public required string Product {  get; set; }
        public required decimal Version { get; set; }
        public required string OS { get; set; }
        public required string Status { get; set; }
        public required string Description { get; set; }
        public required DateTime StartDate { get; set; }
    }
```

Call the stored procedure in C# as follows:

```
            List<Issue> issues = new List<Issue>();
            string connectionstring =
"Server=(localdb)\\mssqllocaldb;Database=Stoneware;Trusted_Connection=True";
            using(var connection = new SqlConnection(connectionstring))
            {
                using(SqlCommand command = new
SqlCommand("GetOutstandingSingleProductSingleVersion", connection))
                {
                    connection.Open();
                    command.CommandType = CommandType.StoredProcedure;
                    //add Parameters
                    command.Parameters.AddWithValue("@ProductName", "Day Trader Wannabe");
                    command.Parameters.AddWithValue("@Version", 1.0);
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            issues.Add(new Issue
                            {
                                TicketId = (decimal)reader["IssueID"],
                                Product = (string)reader["ProductName"],
                                Version = (decimal)reader["VersionID"],
                                OS = (string)reader["OSName"],
                                Status = (string)reader["StatusType"],
                                Description = (string)reader["Description"],
                                StartDate = (DateTime)reader["StartDate"]
                            });
                        }
                    }
                    connection.Close();
                }
            }
```

Author: Cheron Fletcher
Date: 4/20/24

Description: Get all resolved issues for a product (single version)

Stored Procedure Name: `GetResolvedSingleProductSingleVersion`

Parameters:
@ProductName (as string)
@Version (as decimal)

Returns:
A List of Issues, or none if no matches are found.

Uses the following "Issue" Model:

```
    public class Issue
    {
        public required decimal TicketId { get; set; }
        public required string Product {  get; set; }
        public required decimal Version { get; set; }
        public required string OS { get; set; }
        public required string Status { get; set; }
        public required string Description { get; set; }
        public required string Resolution { get; set; }
        public required DateTime StartDate { get; set; }
        public required DateTime EndDate { get; set; }
    }
```

Call the stored procedure in C# as follows:

```
            List<Issue> issues = new List<Issue>();
            string connectionstring =
"Server=(localdb)\\mssqllocaldb;Database=Stoneware;Trusted_Connection=True";
            using(var connection = new SqlConnection(connectionstring))
            {
                using(SqlCommand command = new
SqlCommand("GetResolvedSingleProductSingleVersion", connection))
                {
                    connection.Open();
                    command.CommandType = CommandType.StoredProcedure;
                    //add Parameters
                    command.Parameters.AddWithValue("@ProductName", "Day Trader Wannabe");
                    command.Parameters.AddWithValue("@Version", 1.0);
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            issues.Add(new Issue
                            {
                                TicketId = (decimal)reader["IssueID"],
                                Product = (string)reader["ProductName"],
                                Version = (decimal)reader["VersionID"],
                                OS = (string)reader["OSName"],
                                Status = (string)reader["StatusType"],
                                Description = (string)reader["Description"],
                                Resolution = (string)reader["Resolution"],
                                StartDate = (DateTime)reader["StartDate"],
                                EndDate = (DateTime)reader["EndDate"]
                            });
                        }
                    }
                    connection.Close();
                }
            }
```

Author: Cheron Fletcher
Date: 4/20/24

Description: Get all outstanding issues within date range for a product (all versions)

Stored Procedure Name: `GetOutstandingSingleProductInDateRange`

Parameters:
@ProductName (as string)
@Date1 (as DateOnly)
@Date2 (as DateOnly)

Returns:
A List of Issues, or none if no matches are found.

Uses the following "Issue" Model:

```
    public class Issue
    {
        public required decimal TicketId { get; set; }
        public required string Product {  get; set; }
        public required decimal Version { get; set; }
        public required string OS { get; set; }
        public required string Status { get; set; }
        public required string Description { get; set; }
        public required DateTime StartDate { get; set; }
    }
```

Call the stored procedure in C# as follows:

```
            List<Issue> issues = new List<Issue>();
            string connectionstring =
"Server=(localdb)\\mssqllocaldb;Database=Stoneware;Trusted_Connection=True";
            using(var connection = new SqlConnection(connectionstring))
            {
                using(SqlCommand command = new
SqlCommand("GetOutstandingSingleProductInDateRange", connection))
                {
                    connection.Open();
                    command.CommandType = CommandType.StoredProcedure;
                    //add Parameters
                    command.Parameters.AddWithValue("@ProductName", "Day Trader Wannabe");
                    command.Parameters.AddWithValue("@Date1", new DateOnly(2024, 01, 01));
                    command.Parameters.AddWithValue("@Date2", new DateOnly(2024, 05, 12));
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            issues.Add(new Issue
                            {
                                TicketId = (decimal)reader["IssueID"],
                                Product = (string)reader["ProductName"],
                                Version = (decimal)reader["VersionID"],
                                OS = (string)reader["OSName"],
                                Status = (string)reader["StatusType"],
                                Description = (string)reader["Description"],
                                StartDate = (DateTime)reader["StartDate"]
                            });
                        }
                    }
                    connection.Close();
                }
            }
```

Author: Cheron Fletcher
Date: 4/20/24

Description: Get all resolved issues within date range for a product (all versions)

Stored Procedure Name: `GetResolvedSingleProductInDateRange`

Parameters:
@ProductName (as string)
@Date1 (as DateOnly)
@Date2 (as DateOnly)

Returns:
A List of Issues, or none if no matches are found.

Uses the following "Issue" Model:

```
    public class Issue
    {
        public required decimal TicketId { get; set; }
        public required string Product {  get; set; }
        public required decimal Version { get; set; }
        public required string OS { get; set; }
        public required string Status { get; set; }
        public required string Description { get; set; }
        public required string Resolution { get; set; }
        public required DateTime StartDate { get; set; }
        public required DateTime EndDate { get; set; }
    }
```

Call the stored procedure in C# as follows:

```
            List<Issue> issues = new List<Issue>();
            string connectionstring =
"Server=(localdb)\\mssqllocaldb;Database=Stoneware;Trusted_Connection=True";
            using(var connection = new SqlConnection(connectionstring))
            {
                using(SqlCommand command = new SqlCommand("GetResolvedSingleProductInDateRange",
connection))
                {
                    connection.Open();
                    command.CommandType = CommandType.StoredProcedure;
                    //add Parameters
                    command.Parameters.AddWithValue("@ProductName", "Day Trader Wannabe");
                    command.Parameters.AddWithValue("@Date1", new DateOnly(2024, 01, 01));
                    command.Parameters.AddWithValue("@Date2", new DateOnly(2024, 05, 12));
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            issues.Add(new Issue
                            {
                                TicketId = (decimal)reader["IssueID"],
                                Product = (string)reader["ProductName"],
                                Version = (decimal)reader["VersionID"],
                                OS = (string)reader["OSName"],
                                Status = (string)reader["StatusType"],
                                Description = (string)reader["Description"],
                                Resolution = (string)reader["Resolution"],
                                StartDate = (DateTime)reader["StartDate"],
                                EndDate = (DateTime)reader["EndDate"]
                            });
                        }
                    }
                    connection.Close();
                }
            }
```

Author: Cheron Fletcher
Date: 4/20/24

Description: Get all outstanding issues within date range for a product (single version)

Stored Procedure Name: `GetOutstandingSingleProductSingleVersionInDateRange`

Parameters:
@ProductName (as string)
@Version (as decimal)
@Date1 (as DateOnly)
@Date2 (as DateOnly)

Returns:
A List of Issues, or none if no matches are found.

Uses the following "Issue" Model:

```
    public class Issue
    {
        public required decimal TicketId { get; set; }
        public required string Product {  get; set; }
        public required decimal Version { get; set; }
        public required string OS { get; set; }
        public required string Status { get; set; }
        public required string Description { get; set; }
        public required DateTime StartDate { get; set; }
    }
```

Call the stored procedure in C# as follows:

```
            List<Issue> issues = new List<Issue>();
            string connectionstring =
"Server=(localdb)\\mssqllocaldb;Database=Stoneware;Trusted_Connection=True";
            using(var connection = new SqlConnection(connectionstring))
            {
                using(SqlCommand command = new
SqlCommand("GetOutstandingSingleProductSingleVersionInDateRange", connection))
                {
                    connection.Open();
                    command.CommandType = CommandType.StoredProcedure;
                    //add Parameters
                    command.Parameters.AddWithValue("@ProductName", "Day Trader Wannabe");
                    command.Parameters.AddWithValue("@Version", 1.0);
                    command.Parameters.AddWithValue("@Date1", new DateOnly(2024, 01, 01));
                    command.Parameters.AddWithValue("@Date2", new DateOnly(2024, 05, 12));
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            issues.Add(new Issue
                            {
                                TicketId = (decimal)reader["IssueID"],
                                Product = (string)reader["ProductName"],
                                Version = (decimal)reader["VersionID"],
                                OS = (string)reader["OSName"],
                                Status = (string)reader["StatusType"],
                                Description = (string)reader["Description"],
                                StartDate = (DateTime)reader["StartDate"]
                            });
                        }
                    }
                    connection.Close();
                }
            }
```

Author: Cheron Fletcher
Date: 4/20/24

Description: Get all resolved issues within date range for a product (single version)

Stored Procedure Name: `GetResolvedSingleProductSingleVersionInDateRange`

Parameters:
@ProductName (as string)
@Version (as decimal)
@Date1 (as DateOnly)
@Date2 (as DateOnly)

Returns:
A List of Issues, or none if no matches are found.

Uses the following "Issue" Model:

```
    public class Issue
    {
        public required decimal TicketId { get; set; }
        public required string Product {  get; set; }
        public required decimal Version { get; set; }
        public required string OS { get; set; }
        public required string Status { get; set; }
        public required string Description { get; set; }
        public required string Resolution { get; set; }
        public required DateTime StartDate { get; set; }
        public required DateTime EndDate { get; set; }
    }
```

Call the stored procedure in C# as follows:

```
            List<Issue> issues = new List<Issue>();
            string connectionstring =
"Server=(localdb)\\mssqllocaldb;Database=Stoneware;Trusted_Connection=True";
            using(var connection = new SqlConnection(connectionstring))
            {
                using(SqlCommand command = new
SqlCommand("GetResolvedSingleProductSingleVersionInDateRange", connection))
                {
                    connection.Open();
                    command.CommandType = CommandType.StoredProcedure;
                    //add Parameters
                    command.Parameters.AddWithValue("@ProductName", "Day Trader Wannabe");
                    command.Parameters.AddWithValue("@Version", 1.0);
                    command.Parameters.AddWithValue("@Date1", new DateOnly(2024, 01, 01));
                    command.Parameters.AddWithValue("@Date2", new DateOnly(2024, 05, 12));
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            issues.Add(new Issue
                            {
                                TicketId = (decimal)reader["IssueID"],
                                Product = (string)reader["ProductName"],
                                Version = (decimal)reader["VersionID"],
                                OS = (string)reader["OSName"],
                                Status = (string)reader["StatusType"],
                                Description = (string)reader["Description"],
                                Resolution = (string)reader["Resolution"],
                                StartDate = (DateTime)reader["StartDate"],
                                EndDate = (DateTime)reader["EndDate"]
                            });
                        }
                    }
                    connection.Close();
                }
            }
```

Author: Cheron Fletcher
Date: 4/20/24

Description: Get all outstanding issues containing list of keywords (all products)

Stored Procedure Name: `GetOutstandingWithKeywords`

Parameters:
@Keyword1 (as wildcard, such as: “%price%”)
@Keyword2 (as wildcard, such as: “%trade%”)
@Keyword3 (as wildcard, such as: “%slow%”)
@Keyword4 (as wildcard, such as: “%market%”)

Returns:
A List of Issues, or none if no matches are found.

Uses the following “Issue” Model:

```
    public class Issue
    {
        public required decimal TicketId { get; set; }
        public required string Product {  get; set; }
        public required decimal Version { get; set; }
        public required string OS { get; set; }
        public required string Status { get; set; }
        public required string Description { get; set; }
        public required DateTime StartDate { get; set; }
    }
```

Call the stored procedure in C# as follows:

```
            List<Issue> issues = new List<Issue>();
            string connectionstring =
"Server=(localdb)\\mssqllocaldb;Database=Stoneware;Trusted_Connection=True";
            using(var connection = new SqlConnection(connectionstring))
            {
                using(SqlCommand command = new SqlCommand("GetOutstandingWithKeywords",
connection))
                {
                    connection.Open();
                    command.CommandType = CommandType.StoredProcedure;
                    //add Parameters
                    command.Parameters.AddWithValue("@Keyword1", "%price%");
                    command.Parameters.AddWithValue("@Keyword2", "%trade%");
                    command.Parameters.AddWithValue("@Keyword3", "%slow%");
                    command.Parameters.AddWithValue("@Keyword4", "%market%");
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            issues.Add(new Issue
                            {
                                TicketId = (decimal)reader["IssueID"],
                                Product = (string)reader["ProductName"],
                                Version = (decimal)reader["VersionID"],
                                OS = (string)reader["OSName"],
                                Status = (string)reader["StatusType"],
                                Description = (string)reader["Description"],
                                StartDate = (DateTime)reader["StartDate"]
                            });
                        }
                    }
                    connection.Close();
                }
            }
```

Author: Cheron Fletcher
Date: 4/20/24

Description: Get all resolved issues containing list of keywords (all products)

Stored Procedure Name: `GetResolvedWithKeywords`

Parameters:
@Keyword1 (as wildcard, such as: “%price%”)
@Keyword2 (as wildcard, such as: “%trade%”)
@Keyword3 (as wildcard, such as: “%slow%”)
@Keyword4 (as wildcard, such as: “%market%”)

Returns:
A List of Issues, or none if no matches are found.

Uses the following “Issue” Model:

```
    public class Issue
    {
        public required decimal TicketId { get; set; }
        public required string Product {  get; set; }
        public required decimal Version { get; set; }
        public required string OS { get; set; }
        public required string Status { get; set; }
        public required string Description { get; set; }
        public required string Resolution { get; set; }
        public required DateTime StartDate { get; set; }
        public required DateTime EndDate { get; set; }
    }
```

Call the stored procedure in C# as follows:

```
            List<Issue> issues = new List<Issue>();
            string connectionstring =
"Server=(localdb)\\mssqllocaldb;Database=Stoneware;Trusted_Connection=True";
            using(var connection = new SqlConnection(connectionstring))
            {
                using(SqlCommand command = new SqlCommand("GetResolvedWithKeywords", connection))
                {
                    connection.Open();
                    command.CommandType = CommandType.StoredProcedure;
                    //add Parameters
                    command.Parameters.AddWithValue("@Keyword1", "%price%");
                    command.Parameters.AddWithValue("@Keyword2", "%trade%");
                    command.Parameters.AddWithValue("@Keyword3", "%slow%");
                    command.Parameters.AddWithValue("@Keyword4", "%market%");
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            issues.Add(new Issue
                            {
                                TicketId = (decimal)reader["IssueID"],
                                Product = (string)reader["ProductName"],
                                Version = (decimal)reader["VersionID"],
                                OS = (string)reader["OSName"],
                                Status = (string)reader["StatusType"],
                                Description = (string)reader["Description"],
                                Resolution = (string)reader["Resolution"],
                                StartDate = (DateTime)reader["StartDate"],
                                EndDate = (DateTime)reader["EndDate"]
                            });
                        }
                    }
                    connection.Close();
                }
            }
```

Author: Cheron Fletcher
Date: 4/20/24

Description: Get all outstanding issues for a product containing list of keywords (all versions)

Stored Procedure Name: `GetOutstandingSingleProductWithKeywords`

Parameters:
@ProductName (as string)
@Keyword1 (as wildcard, such as: “%price%”)
@Keyword2 (as wildcard, such as: “%trade%”)
@Keyword3 (as wildcard, such as: “%slow%”)
@Keyword4 (as wildcard, such as: “%market%”)

Returns:
A List of Issues, or none if no matches are found.

Uses the following “Issue” Model:

```
    public class Issue
    {
        public required decimal TicketId { get; set; }
        public required string Product {  get; set; }
        public required decimal Version { get; set; }
        public required string OS { get; set; }
        public required string Status { get; set; }
        public required string Description { get; set; }
        public required DateTime StartDate { get; set; }
    }
```

Call the stored procedure in C# as follows:

```
            List<Issue> issues = new List<Issue>();
            string connectionstring =
"Server=(localdb)\\mssqllocaldb;Database=Stoneware;Trusted_Connection=True";
            using(var connection = new SqlConnection(connectionstring))
            {
                using(SqlCommand command = new
SqlCommand("GetOutstandingSingleProductWithKeywords", connection))
                {
                    connection.Open();
                    command.CommandType = CommandType.StoredProcedure;
                    //add Parameters
                    command.Parameters.AddWithValue("@ProductName", "Day Trader Wannabe");
                    command.Parameters.AddWithValue("@Keyword1", "%price%");
                    command.Parameters.AddWithValue("@Keyword2", "%trade%");
                    command.Parameters.AddWithValue("@Keyword3", "%slow%");
                    command.Parameters.AddWithValue("@Keyword4", "%market%");
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            issues.Add(new Issue
                            {
                                TicketId = (decimal)reader["IssueID"],
                                Product = (string)reader["ProductName"],
                                Version = (decimal)reader["VersionID"],
                                OS = (string)reader["OSName"],
                                Status = (string)reader["StatusType"],
                                Description = (string)reader["Description"],
                                StartDate = (DateTime)reader["StartDate"]
                            });
                        }
                    }
                    connection.Close();
                }
            }
```

Author: Cheron Fletcher
Date: 4/20/24

Description: Get all resolved issues for a product containing list of keywords (all versions)

Stored Procedure Name: `GetResolvedSingleProductWithKeywords`

Parameters:
@ProductName (as string)
@Keyword1 (as wildcard, such as: “%price%”)
@Keyword2 (as wildcard, such as: “%trade%”)
@Keyword3 (as wildcard, such as: “%slow%”)
@Keyword4 (as wildcard, such as: “%market%”)

Returns:
A List of Issues, or none if no matches are found.

Uses the following “Issue” Model:

```
    public class Issue
    {
        public required decimal TicketId { get; set; }
        public required string Product {  get; set; }
        public required decimal Version { get; set; }
        public required string OS { get; set; }
        public required string Status { get; set; }
        public required string Description { get; set; }
        public required string Resolution { get; set; }
        public required DateTime StartDate { get; set; }
        public required DateTime EndDate { get; set; }
    }
```

Call the stored procedure in C# as follows:

```
            List<Issue> issues = new List<Issue>();
            string connectionstring =
"Server=(localdb)\\mssqllocaldb;Database=Stoneware;Trusted_Connection=True";
            using(var connection = new SqlConnection(connectionstring))
            {
                using(SqlCommand command = new SqlCommand("GetResolvedSingleProductWithKeywords",
connection))
                {
                    connection.Open();
                    command.CommandType = CommandType.StoredProcedure;
                    //add Parameters
                    command.Parameters.AddWithValue("@ProductName", "Day Trader Wannabe");
                    command.Parameters.AddWithValue("@Keyword1", "%price%");
                    command.Parameters.AddWithValue("@Keyword2", "%trade%");
                    command.Parameters.AddWithValue("@Keyword3", "%slow%");
                    command.Parameters.AddWithValue("@Keyword4", "%market%");
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            issues.Add(new Issue
                            {
                                TicketId = (decimal)reader["IssueID"],
                                Product = (string)reader["ProductName"],
                                Version = (decimal)reader["VersionID"],
                                OS = (string)reader["OSName"],
                                Status = (string)reader["StatusType"],
                                Description = (string)reader["Description"],
                                Resolution = (string)reader["Resolution"],
                                StartDate = (DateTime)reader["StartDate"],
                                EndDate = (DateTime)reader["EndDate"]
                            });
                        }
                    }
                    connection.Close();
                }
            }
```

Author: Cheron Fletcher
Date: 4/20/24

Description: Get all outstanding issues for a product containing list of keywords (single version)

Stored Procedure Name: `GetOutstandingSingleProductSingleVersionWithKeywords`

Parameters:
@ProductName (as string)
@Version (as decimal)
@Keyword1 (as wildcard, such as: “%price%”)
@Keyword2 (as wildcard, such as: “%trade%”)
@Keyword3 (as wildcard, such as: “%slow%”)
@Keyword4 (as wildcard, such as: “%market%”)

Returns:
A List of Issues, or none if no matches are found.

Uses the following “Issue” Model:

```
    public class Issue
    {
        public required decimal TicketId { get; set; }
        public required string Product {  get; set; }
        public required decimal Version { get; set; }
        public required string OS { get; set; }
        public required string Status { get; set; }
        public required string Description { get; set; }
        public required DateTime StartDate { get; set; }
    }
```

Call the stored procedure in C# as follows:

```
            List<Issue> issues = new List<Issue>();
            string connectionstring =
"Server=(localdb)\\mssqllocaldb;Database=Stoneware;Trusted_Connection=True";
            using(var connection = new SqlConnection(connectionstring))
            {
                using(SqlCommand command = new
SqlCommand("GetOutstandingSingleProductSingleVersionWithKeywords", connection))
                {
                    connection.Open();
                    command.CommandType = CommandType.StoredProcedure;
                    //add Parameters
                    command.Parameters.AddWithValue("@ProductName", "Day Trader Wannabe");
                    command.Parameters.AddWithValue("@Version", 1.0);
                    command.Parameters.AddWithValue("@Keyword1", "%price%");
                    command.Parameters.AddWithValue("@Keyword2", "%trade%");
                    command.Parameters.AddWithValue("@Keyword3", "%slow%");
                    command.Parameters.AddWithValue("@Keyword4", "%market%");
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            issues.Add(new Issue
                            {
                                TicketId = (decimal)reader["IssueID"],
                                Product = (string)reader["ProductName"],
                                Version = (decimal)reader["VersionID"],
                                OS = (string)reader["OSName"],
                                Status = (string)reader["StatusType"],
                                Description = (string)reader["Description"],
                                StartDate = (DateTime)reader["StartDate"]
                            });
                        }
                    }
                    connection.Close();
                }
            }
```

Author: Cheron Fletcher
Date: 4/20/24

Description: Get all resolved issues for a product containing list of keywords (single version)

Stored Procedure Name: `GetResolvedSingleProductSingleVersionWithKeywords`

Parameters:
@ProductName (as string)
@Version (as decimal)
@Keyword1 (as wildcard, such as: “%price%”)
@Keyword2 (as wildcard, such as: “%trade%”)
@Keyword3 (as wildcard, such as: “%slow%”)
@Keyword4 (as wildcard, such as: “%market%”)

Returns:
A List of Issues, or none if no matches are found.

Uses the following “Issue” Model:

```
    public class Issue
    {
        public required decimal TicketId { get; set; }
        public required string Product {  get; set; }
        public required decimal Version { get; set; }
        public required string OS { get; set; }
        public required string Status { get; set; }
        public required string Description { get; set; }
        public required string Resolution { get; set; }
        public required DateTime StartDate { get; set; }
        public required DateTime EndDate { get; set; }
    }
```

Call the stored procedure in C# as follows:

```
            List<Issue> issues = new List<Issue>();
            string connectionstring =
"Server=(localdb)\\mssqllocaldb;Database=Stoneware;Trusted_Connection=True";
            using(var connection = new SqlConnection(connectionstring))
            {
                using(SqlCommand command = new
SqlCommand("GetResolvedSingleProductSingleVersionWithKeywords", connection))
                {
                    connection.Open();
                    command.CommandType = CommandType.StoredProcedure;
                    //add Parameters
                    command.Parameters.AddWithValue("@ProductName", "Day Trader Wannabe");
                    command.Parameters.AddWithValue("@Version", 1.0);
                    command.Parameters.AddWithValue("@Keyword1", "%price%");
                    command.Parameters.AddWithValue("@Keyword2", "%trade%");
                    command.Parameters.AddWithValue("@Keyword3", "%slow%");
                    command.Parameters.AddWithValue("@Keyword4", "%market%");
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            issues.Add(new Issue
                            {
                                TicketId = (decimal)reader["IssueID"],
                                Product = (string)reader["ProductName"],
                                Version = (decimal)reader["VersionID"],
                                OS = (string)reader["OSName"],
                                Status = (string)reader["StatusType"],
                                Description = (string)reader["Description"],
                                Resolution = (string)reader["Resolution"],
                                StartDate = (DateTime)reader["StartDate"],
                                EndDate = (DateTime)reader["EndDate"]
                            });
                        }
                    }
                    connection.Close();
                }
            }
```

Author: Cheron Fletcher
Date: 4/20/24

Description: Get all outstanding issues within date range for a product containing list of keywords (all versions)

Stored Procedure Name: `GetOutstandingSingleProductInDateRangeWithKeywords`

Parameters:
@ProductName (as string)
@Date1 (as DateOnly)
@Date2 (as DateOnly)
@Keyword1 (as wildcard, such as: “%price%”)
@Keyword2 (as wildcard, such as: “%trade%”)
@Keyword3 (as wildcard, such as: “%slow%”)
@Keyword4 (as wildcard, such as: “%market%”)

Returns:
A List of Issues, or none if no matches are found.

Uses the following “Issue” Model:

```
    public class Issue
    {
        public required decimal TicketId { get; set; }
        public required string Product {  get; set; }
        public required decimal Version { get; set; }
        public required string OS { get; set; }
        public required string Status { get; set; }
        public required string Description { get; set; }
        public required DateTime StartDate { get; set; }
    }
```

Call the stored procedure in C# as follows:

```
            List<Issue> issues = new List<Issue>();
            string connectionstring =
"Server=(localdb)\\mssqllocaldb;Database=Stoneware;Trusted_Connection=True";
            using(var connection = new SqlConnection(connectionstring))
            {
                using(SqlCommand command = new
SqlCommand("GetOutstandingSingleProductInDateRangeWithKeywords", connection))
                {
                    connection.Open();
                    command.CommandType = CommandType.StoredProcedure;
                    //add Parameters
                    command.Parameters.AddWithValue("@ProductName", "Day Trader Wannabe");
                    command.Parameters.AddWithValue("@Date1", new DateOnly(2024, 01, 01));
                    command.Parameters.AddWithValue("@Date2", new DateOnly(2024, 05, 12));
                    command.Parameters.AddWithValue("@Keyword1", "%price%");
                    command.Parameters.AddWithValue("@Keyword2", "%trade%");
                    command.Parameters.AddWithValue("@Keyword3", "%slow%");
                    command.Parameters.AddWithValue("@Keyword4", "%market%");
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            issues.Add(new Issue
                            {
                                TicketId = (decimal)reader["IssueID"],
                                Product = (string)reader["ProductName"],
                                Version = (decimal)reader["VersionID"],
                                OS = (string)reader["OSName"],
                                Status = (string)reader["StatusType"],
                                Description = (string)reader["Description"],
                                StartDate = (DateTime)reader["StartDate"]
                            });
                        }
                    }
                    connection.Close();
                }
            }
```

Author: Cheron Fletcher
Date: 4/20/24

Description: Get all resolved issues within date range for a product containing list of keywords (all versions)

Stored Procedure Name: `GetResolvedSingleProductInDateRangeWithKeywords`

Parameters:
@ProductName (as string)
@Date1 (as DateOnly)
@Date2 (as DateOnly)
@Keyword1 (as wildcard, such as: “%price%”)
@Keyword2 (as wildcard, such as: “%trade%”)
@Keyword3 (as wildcard, such as: “%slow%”)
@Keyword4 (as wildcard, such as: “%market%”)

Returns:
A List of Issues, or none if no matches are found.

Uses the following “Issue” Model:

```
    public class Issue
    {
        public required decimal TicketId { get; set; }
        public required string Product {  get; set; }
        public required decimal Version { get; set; }
        public required string OS { get; set; }
        public required string Status { get; set; }
        public required string Description { get; set; }
        public required string Resolution { get; set; }
        public required DateTime StartDate { get; set; }
        public required DateTime EndDate { get; set; }
    }
```

Call the stored procedure in C# as follows:

```
            List<Issue> issues = new List<Issue>();
            string connectionstring =
"Server=(localdb)\\mssqllocaldb;Database=Stoneware;Trusted_Connection=True";
            using(var connection = new SqlConnection(connectionstring))
            {
                using(SqlCommand command = new
SqlCommand("GetResolvedSingleProductInDateRangeWithKeywords", connection))
                {
                    connection.Open();
                    command.CommandType = CommandType.StoredProcedure;
                    //add Parameters
                    command.Parameters.AddWithValue("@ProductName", "Day Trader Wannabe");
                    command.Parameters.AddWithValue("@Date1", new DateOnly(2024, 01, 01));
                    command.Parameters.AddWithValue("@Date2", new DateOnly(2024, 05, 12));
                    command.Parameters.AddWithValue("@Keyword1", "%price%");
                    command.Parameters.AddWithValue("@Keyword2", "%trade%");
                    command.Parameters.AddWithValue("@Keyword3", "%slow%");
                    command.Parameters.AddWithValue("@Keyword4", "%market%");
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            issues.Add(new Issue
                            {
                                TicketId = (decimal)reader["IssueID"],
                                Product = (string)reader["ProductName"],
                                Version = (decimal)reader["VersionID"],
                                OS = (string)reader["OSName"],
                                Status = (string)reader["StatusType"],
                                Description = (string)reader["Description"],
                                Resolution = (string)reader["Resolution"],
                                StartDate = (DateTime)reader["StartDate"],
                                EndDate = (DateTime)reader["EndDate"]
                            });
                        }
                    }
                    connection.Close();
                }
            }
```

Author: Cheron Fletcher
Date: 4/20/24

Description: Gets all outstanding issues within date range for a product containing list of keywords (single version)

Stored Procedure Name:
`GetOutstandingSingleProductSingleVersionInDateRangeWithKeywords`

Parameters:
@ProductName (as string)
@Version (as decimal)
@Date1 (as DateOnly)
@Date2 (as DateOnly)
@Keyword1 (as wildcard, such as: “%price%”)
@Keyword2 (as wildcard, such as: “%trade%”)
@Keyword3 (as wildcard, such as: “%slow%”)
@Keyword4 (as wildcard, such as: “%market%”)

Returns:
A List of Issues, or none if no matches are found.

Uses the following “Issue” Model:

```
    public class Issue
    {
        public required decimal TicketId { get; set; }
        public required string Product {  get; set; }
        public required decimal Version { get; set; }
        public required string OS { get; set; }
        public required string Status { get; set; }
        public required string Description { get; set; }
        public required DateTime StartDate { get; set; }
    }
```

Call the stored procedure in C# as follows:

```
            List<Issue> issues = new List<Issue>();
            string connectionstring =
"Server=(localdb)\\mssqllocaldb;Database=Stoneware;Trusted_Connection=True";
            using(var connection = new SqlConnection(connectionstring))
            {
                using(SqlCommand command = new
SqlCommand("GetOutstandingSingleProductSingleVersionInDateRangeWithKeywords", connection))
                {
                    connection.Open();
                    command.CommandType = CommandType.StoredProcedure;
                    //add Parameters
                    command.Parameters.AddWithValue("@ProductName", "Day Trader Wannabe");
                    command.Parameters.AddWithValue("@Version", 1.0);
                    command.Parameters.AddWithValue("@Date1", new DateOnly(2024, 01, 01));
                    command.Parameters.AddWithValue("@Date2", new DateOnly(2024, 05, 12));
                    command.Parameters.AddWithValue("@Keyword1", "%price%");
                    command.Parameters.AddWithValue("@Keyword2", "%trade%");
                    command.Parameters.AddWithValue("@Keyword3", "%slow%");
                    command.Parameters.AddWithValue("@Keyword4", "%market%");
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            issues.Add(new Issue
                            {
                                TicketId = (decimal)reader["IssueID"],
                                Product = (string)reader["ProductName"],
                                Version = (decimal)reader["VersionID"],
                                OS = (string)reader["OSName"],
                                Status = (string)reader["StatusType"],
                                Description = (string)reader["Description"],
                                StartDate = (DateTime)reader["StartDate"]
                            });
                        }
                    }
                    connection.Close();
                }
            }
```

Author: Cheron Fletcher
Date: 4/20/24

Description: Gets all resolved issues within date range for a product containing list of keywords (single version)

Stored Procedure Name:
`GetResolvedSingleProductSingleVersionInDateRangeWithKeywords`

Parameters:
@ProductName (as string)
@Version (as decimal)
@Date1 (as DateOnly)
@Date2 (as DateOnly)
@Keyword1 (as wildcard, such as: “%price%”)
@Keyword2 (as wildcard, such as: “%trade%”)
@Keyword3 (as wildcard, such as: “%slow%”)
@Keyword4 (as wildcard, such as: “%market%”)

Returns:
A List of Issues, or none if no matches are found.

Uses the following “Issue” Model:

```
    public class Issue
    {
        public required decimal TicketId { get; set; }
        public required string Product {  get; set; }
        public required decimal Version { get; set; }
        public required string OS { get; set; }
        public required string Status { get; set; }
        public required string Description { get; set; }
        public required string Resolution { get; set; }
        public required DateTime StartDate { get; set; }
        public required DateTime EndDate { get; set; }
    }
```

Call the stored procedure in C# as follows:

```
            List<Issue> issues = new List<Issue>();
            string connectionstring =
"Server=(localdb)\\mssqllocaldb;Database=Stoneware;Trusted_Connection=True";
            using(var connection = new SqlConnection(connectionstring))
            {
                using(SqlCommand command = new
SqlCommand("GetResolvedSingleProductSingleVersionInDateRangeWithKeywords", connection))
                {
                    connection.Open();
                    command.CommandType = CommandType.StoredProcedure;
                    //add Parameters
                    command.Parameters.AddWithValue("@ProductName", "Day Trader Wannabe");
                    command.Parameters.AddWithValue("@Version", 1.0);
                    command.Parameters.AddWithValue("@Date1", new DateOnly(2024, 01, 01));
                    command.Parameters.AddWithValue("@Date2", new DateOnly(2024, 05, 12));
                    command.Parameters.AddWithValue("@Keyword1", "%price%");
                    command.Parameters.AddWithValue("@Keyword2", "%trade%");
                    command.Parameters.AddWithValue("@Keyword3", "%slow%");
                    command.Parameters.AddWithValue("@Keyword4", "%market%");
                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            issues.Add(new Issue
                            {
                                TicketId = (decimal)reader["IssueID"],
                                Product = (string)reader["ProductName"],
                                Version = (decimal)reader["VersionID"],
                                OS = (string)reader["OSName"],
                                Status = (string)reader["StatusType"],
                                Description = (string)reader["Description"],
                                Resolution = (string)reader["Resolution"],
                                StartDate = (DateTime)reader["StartDate"],
                                EndDate = (DateTime)reader["EndDate"]
                            });
                        }
                    }
                    connection.Close();
                }
            }
```