[MS-APPMDP]:
SharePoint App Management Database Protocol Specification

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft Open Specification Promise or the Community Promise. If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights.
- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

Reservation of Rights. All other rights are reserved, and this notice does not grant any rights other than specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications do not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments you are free to take advantage of them. Certain Open Specifications are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

Preliminary Documentation. This Open Specification provides documentation for past and current releases and/or for the pre-release (beta) version of this technology. This Open Specification is final documentation for past or current releases as specifically noted in the document, as applicable; it is preliminary documentation for the pre-release (beta) versions. Microsoft will release final documentation in connection with the commercial release of the updated or new version of this technology. As the documentation may change between this preliminary version and the final version of this technology, there are risks in relying on preliminary documentation. To the extent that you incur additional development obligations or any other costs as a result of relying on this preliminary documentation, you do so at your own risk.

Revision Summary

Date	Revision History	Revision Class	Comments
01/20/2012	0.1	New	Released new document.
04/11/2012	0.1	No change	No changes to the meaning, language, or formatting of the technical content.
07/16/2012	0.1	No change	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction

The SharePoint App Management Database Protocol enables a protocol client to store and retrieve information used to manage registration, permissions and licenses of **apps**.

Sections 1.8, 2, and 3 of this specification are normative and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in RFC 2119. Sections 1.5 and 1.9 are also normative but cannot contain those terms. All other sections and examples in this specification are informative.

1.1 Glossary

The following terms are defined in [MS-GLOS]:

GUID
UTF-16

The following terms are defined in [MS-OFCGLOS]:

app
app instance
app principal
app product identifier
data range
database application
farm
marketplace asset identifier
marketplace license
marketplace license token
result set
return code
security principal
site subscription
site subscription identifier
stored procedure
Uniform Resource Identifier (URI)
Uniform Resource Locator (URL)
XML schema

The following terms are specific to this document:

app principal permission provider: An entity that grants app principal permissions.

app principal realm: A Uniform Resource Identifier (URI) that is the source of requests from the app principal.

marketplace app provider: The name of the individual or company that submitted the app to the marketplace.

marketplace billing market: A specifier that identifies the region of the credit card used to buy apps.

marketplace content market: A collection of metadata associated with each individual app that contains information such as the region and locale of the app.

marketplace deployment identifier: The identifier of the location to which app licenses are sold.

marketplace license director: An administrator who can grant marketplace user assignments to others.

marketplace license type: A definition of the usage pattern of a marketplace license that can enable trial periods for or perpetual usage of an app, and can also limit the number of people able to use the app.

marketplace license user assignment: The set of rights from a marketplace license given to a specific user.

MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as described in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

1.2 References

References to Microsoft Open Specifications documentation do not include a publishing year because links are to the latest version of the technical documents, which are updated frequently. References to other documents include a publishing year when one is available.

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information. Please check the archive site, http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.

[Iseminger] Microsoft Corporation, "SQL Server 2000 Architecture and XML/Internet Support", Volume 1 of Microsoft SQL Server 2000 Reference Library, Microsoft Press, 2001, ISBN 0-7356-1280-3, http://www.microsoft.com/mspress/books/5001.aspx

[MSDN-TSQL-Ref] Microsoft Corporation, "Transact-SQL Reference", http://msdn.microsoft.com/en-us/library/ms189826(SQL.90).aspx

[MS-SPSSDBSOGP] Microsoft Corporation, "SharePoint Shared Service Database Scale Out Generic Protocol Specification".

[MS-TDS] Microsoft Corporation, "Tabular Data Stream Protocol Specification".

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.rfc-editor.org/rfc/rfc2119.txt

1.2.2 Informative References

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary".

[MS-OFCGLOS] Microsoft Corporation, "Microsoft Office Master Glossary".

1.3 Overview

This protocol provides the protocol client, which works in conjunction with a **database application** acting as a protocol server, the following functionalities:

- Storing and managing basic information for apps.
- Registering **app principals** with associated app principal permission which are used for authorization against various **farm** resources.
- Storing and managing **marketplace licenses**, **marketplace license directors**, **marketplace license user assignments**.
- Applying a marketplace license renewal process.
- Randomly generating, reading and updating **marketplace deployment identifiers** associated with each **site subscription**.
- Maintaining the relationship between app principals, **app instances** and **app product identifiers** on a given site subscription.

1.4 Relationship to Other Protocols

The following diagram shows the transport stack that the protocol uses:

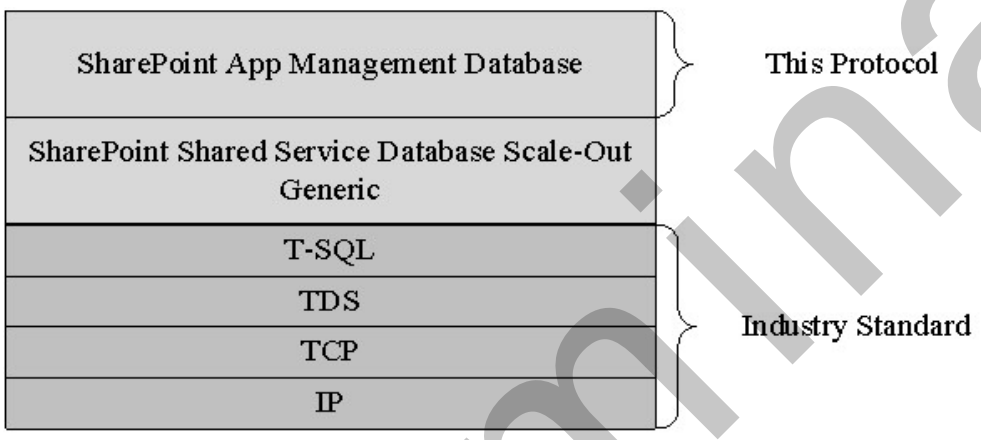

Figure 1: This protocol in relation to other protocols

This protocol implements the SharePoint Shared Service Database Scale-Out Generic protocol with the **PartitionKeySize** value 529 as described in [MS-SPSSDBSOGP] section 1.4.

1.5 Prerequisites/Preconditions

This protocol operates between a protocol client and a protocol server on which the back-end databases are stored. The protocol client is expected to know the location and connection information for the databases.

This protocol requires that the protocol client has appropriate permissions to call the stored procedures in the back-end databases.

1.6 Applicability Statement

This protocol is intended for use by protocol clients and protocol servers that are both connected by high-bandwidth, low latency network connections.

1.7 Versioning and Capability Negotiation

Security and authentication methods: This protocol supports the SSPI and SQL Authentication with the Protocol Server role in [MS-TDS].

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

[MS-TDS] is the transport protocol used to call the stored procedures, **return code** and **result sets**.

2.2 Common Data Types

The following sections define the common data types that are used in this protocol.

2.2.1 Simple Data Types and Enumerations

2.2.1.1 RequestGuid

RequestGuid: uniqueidentifier NULL. A **GUID** uniquely identifying the current request.

2.2.1.2 LicenseId

LicenseId: uniqueidentifier NOT NULL. The identifier of a marketplace license.

2.2.1.3 AssetId

AssetId: nvarchar(14) NOT NULL. Represents a **marketplace asset identifier**.

2.2.1.4 AppName

AppName: nvarchar(1024) NOT NULL. Represents the title of the app for a particular marketplace license.

2.2.1.5 UserIdentity

UserIdentity: nvarchar(255) NOT NULL. Uniquely represents the name of a **security principal (2)**.

2.2.1.6 UserKey

UserKey: nvarchar(255) NULL. Key of the security principal (2) that represents a user to a particular marketplace license.

2.2.1.7 PurchaserSPIdentity

PurchaserSPIdentity: nvarchar(255) NOT NULL. Name of the security principal (2) that represents the purchaser for a particular marketplace license.

2.2.1.8 PurchaserIdentity

PurchaserIdentity: nvarchar(16) NOT NULL. The identifier for the purchaser of a particular marketplace license.

2.2.1.9 LicenseAcquisitionDate

LicenseAcquisitionDate: datetime NOT NULL. Represents the date and time the marketplace license was acquired.

2.2.1.10 AppDate

AppDate: datetime NOT NULL. Represents the appearance date of a marketplace license. If the **LicenseType**(section 2.2.2.2) is equal to 1 or 3 this date MUST be the **LicenseAcquisitionDate**(section 2.2.1.9). If the **LicenseType**(section 2.2.2.2) is equal to 0 or 2 this date MUST be the date the user was assigned to this marketplace license.

2.2.1.11 ExpirationDate

ExpirationDate: datetime NULL. Represents the trial expiration date for trial marketplace licenses.

2.2.1.12 TokenExpiryDate

TokenExpiryDate: datetime NOT NULL. Represents the expiration date of the token of a marketplace license.

2.2.1.13 IsLicenseExpired

IsLicenseExpired: bit NOT NULL. A bit that specifies whether the marketplace license has expired.

2.2.1.14 IsTokenExpired

IsTokenExpired: bit NOT NULL. A bit that specifies whether the token of a marketplace license has expired or whether the marketplace license qualifies for an automatic renewal attempt.

2.2.1.15 MaxUserCount

MaxUserCount: int NOT NULL. Represents the maximum number of users allowed for a particular marketplace license.

2.2.1.16 CurrentUserCount

CurrentUserCount: int NULL. Represents the current number of users assigned in a particular marketplace license.

2.2.1.17 DeploymentId

DeploymentId: uniqueidentifier NOT NULL. Represents a marketplace deployment identifier..

2.2.1.18 AppId

AppId: nvarchar(256) NOT NULL. Represents an App Principal Identifier.

2.2.1.19 StartingRowNumber

StartingRowNumber: int NOT NULL. The row number of the record from which to start returning result records. The value MUST be non-negative.

2.2.1.20 LimitCount

LimitCount: int NOT NULL. The maximum number of records that can be returned. The value MUST be non-negative.

2.2.1.21 PageSize

PageSize: int NOT NULL. The maximum number of records that can be returned. The value MUST be non-negative.

2.2.1.22 RangeStart

RangeStart: varbinary(529) NULL. The start point of a **data range**. (see [MS-SPSSDBSOGP] for partitions, ranges and database splitting).

2.2.1.23 RangeEnd

RangeEnd: varbinary(529) NULL. The end point of a data range (see [MS-SPSSDBSOGP] for partitions, ranges and database splitting).

2.2.1.24 SearchString

SearchString: nvarchar(1024) NULL. A text used for searching items.

2.2.1.25 LicenseDirectorIdentity

LicenseDirectorIdentity: nvarchar(255) NOT NULL. Name of the security principal (2) that represents a marketplace license director.

2.2.1.26 LicenseDirectorKey

LicenseDirectorKey: nvarchar(255) NOT NULL. Key of the security principal (2) that represents a marketplace license director.

2.2.1.27 DeploymentData

DeploymentData: nvarchar(max) NULL. Data about locations where an app instance is usable. Its format is specific to the implementation of the protocol client, and is not known by the protocol server.

2.2.1.28 ContentMarket

ContentMarket: nvarchar(10) NOT NULL. The identifier of the **marketplace content market** of an app.

2.2.1.29 BillingMarket

BillingMarket: nvarchar(2) NOT NULL. The identifier of the **marketplace billing market** of an app.

2.2.1.30 IconUrl

IconUrl: nvarchar(255) NULL. The location of the icon image of an app.

2.2.1.31 ProviderName

ProviderName: nvarchar(255) NOT NULL. The name of the publisher of an app.

2.2.1.32 ProviderTypeId

ProviderTypeId: uniqueidentifier NOT NULL. The identifier of an **app principal permission provider**.

2.2.1.33 RedirectURL

RedirectURL: nvarchar(255) NULL. The redirect URL of an app principal.

2.2.1.34 Realm

Realm: nvarchar(255) NULL. The realm of an app principal.

2.2.1.35 IsUpdateAllowed

IsUpdateAllowed: bit NOT NULL. A bit that specifies whether an item update operation is allowed.

2.2.1.36 AppInstanceIdentifier

AppInstanceIdentifier: uniqueidentifier NULL. The identifier of an app instance.

2.2.1.37 RawXMLEntitlementToken

RawXMLEntitlementToken: nvarchar(512) NOT NULL. The token issued for a marketplace license.

2.2.1.38 ItemCount

ItemCount: int NOT NULL. A non-negative number representing the count of items.

2.2.1.39 AppPrincipalTitle

AppPrincipalTitle: nvarchar(255) NOT NULL. The title of an app principal.

2.2.1.40 Permission

Permission: varbinary(max) NOT NULL. The implementation specific permissions of an app principal.

2.2.1.41 Version

Version: int NOT NULL**.** An integer representing an app principal permission version. This value MUST NOT be negative.

2.2.2 Bit Fields and Flag Structures

2.2.2.1 AppPrincipalFlags

AppPrincipalFlags: int NOT NULL. The implementation specific state of an app principal.

2.2.2.2 LicenseType

LicenseType: tinyint NOT NULL. The **marketplace license type**.

Value	Description
0	PerpetualMultiUser marketplace license type.
1	PerpetualAllUsers marketplace license type.
2	TrialMultiUser marketplace license type.
3	TrialAllUsers marketplace license type.

2.2.2.3 OmexLicenseType

OmexLicenseType: tinyint NOT NULL. The commercial type of the marketplace license, used for commercial purposes and stored in the protocol server.

2.2.2.4 RightsLevel

RightsLevel: tinyint NOT NULL. Represents the level of rights a marketplace license director has on a marketplace license.

Value	Description
0	NoRights rights level.
1	UserAssignments rights level.
2	ManageRights rights level.

2.2.3 Binary Structures

2.2.3.1 AppRegistryCompositePartitionKey

The **AppRegistryCompositePartitionKey** is 529 bytes long. The first 16 bytes consist of an implementation specific representation of a **site subscription identifier**, the 17th byte is the number 0, and the remaining bytes contains an implementation specific representation of an App Principal Identifier, which is a string in **UTF-16** format and has a length of at most 256 characters. Bytes unused by the App Principal Identifier are set to 0.

2.2.3.2 AppLicensingCompositePartitionKey

The **AppLicensingCompositePartitionKey** is 33 bytes long. The first 16 bytes consist of an implementation specific representation of a site subscription identifier, the 17th byte is the number 1, and the remaining bytes consist of an implementation specific representation of an app product identifier.

2.2.3.3 DeploymentIdCompositePartitionKey

The **DeploymentIdCompositePartitionKey** is 17 bytes long. The first 16 bytes consist of an implementation specific representation of a site subscription identifier, and the 17th byte is the number 2.

2.2.4 Result Sets

This section defines common result sets that are used by this protocol specification.

2.2.4.1 Deployment ID

The **Deployment ID** result set contains the marketplace deployment identifiers corresponding to site subscription identifiers.

```
DeploymentId uniqueidentifier,
```

DeploymentId: The marketplace deployment identifier corresponding to a site subscription identifier. The value MUST be a **DeploymentId**(section 2.2.1.17).

2.2.4.2 Mapping Error

The **Mapping Error** result set contains error codes resulting from operations related to the data range of the protocol server.

```
MappingError int,
```

MappingError: The error code. It takes values from the following table:

Value	Description
-1000	The requested operation cannot be performed according to the data range of the protocol server.
0	No errors encountered.

2.2.4.3 Displayable App Information

The **Displayable App Information** result set contains metadata about an app.

```
CompositePartitionKey varbinary(33),
AssetId nvarchar(14),
ContentMarket nvarchar(10),
BillingMarket nvarchar(2),
IconUrl nvarchar(255),
ProviderName nvarchar(255),
```

CompositePartitionKey: The composite partition key identifying the site subscription identifier and the app product identifier. The value MUST be an **AppLicensingCompositePartitionKey**(section 2.2.3.2).

AssetId: The marketplace asset identifier of the app. The value MUST be an **AssetId** (section 2.2.1.3).

ContentMarket: The marketplace content market of the app. The value MUST be a **ContentMarket** (section 2.2.1.28).

BillingMarket: The marketplace billing market of the app. The value MUST be a **BillingMarket** (section 2.2.1.29).

IconUrl: The icon **URL** of the app. The value MUST be an **IconUrl** (section 2.2.1.30).

ProviderName: The name for the **marketplace app provider** of the app. The value MUST be a **ProviderName** (section 2.2.1.31).

2.2.4.4 License Import Results

License Import Results result set contains information about the marketplace license update results of a marketplace license renewal process.

```
CompositePartitionKey varbinary(33),
PurchaserIdentity nvarchar(16),
DeploymentId uniqueidentifier,
ErrorCode int,
```

CompositePartitionKey: The composite partition key identifying the app product identifier and the site subscription identifier for the app of the marketplace license. The value MUST be an **AppLicensingCompositePartitionKey** (section 2.2.3.2).

PurchaserIdentity: The identifier of the user who purchased the marketplace license. This value must be a **PurchaserIdentity** (section 2.2.1.8).

DeploymentId: The marketplace deployment identifier to which this marketplace license is tied. This value MUST be a **DeploymentId** (section 2.2.1.17).

ErrorCode: The resulting error code from the update operation done to a particular marketplace license from the **License List** (section 2.2.7.4). It takes values from the following table:

Value	Description
-16	*@MaxUserCount* is non-positive and *@LicenseType* is equal to 0 or 2.
-10	*@ExpirationDate* is not null and *@LicenseType* is equal to 0 or 1.
-9	*@ExpirationDate* is null and *@LicenseType* is equal to 2 or 3.
-3	*@MaxUserCount* is not null and *@LicenseType* is equal to 1 or 3.
-2	*@MaxUserCount* is null and *@LicenseType* is equal to 0 or 2.
0	No errors encountered.

2.2.4.5 App Principals

The **App Principals** result set contains information about app principals.

```
CompositePartitionKey varbinary(529),
Title nvarchar(255),
RedirectUrl nvarchar(255),
Realm nvarchar(255),
Flag int,
```

CompositePartitionKey: Uniquely identifies an app principal identifier, site subscription identifier pair. The value MUST be an **AppRegistryCompositePartitionKey** (section 2.2.3.1).

Title: The title of the app principal. This value MUST be an **AppPrincipalTitle**(section 2.2.1.39).

RedirectUrl: The app principal redirect **Uniform Resource Identifier (URI)** for this app principal. The value MUST be a **RedirectUrl** (section 2.2.1.33)

Realm: A URI representing the **app principal realm**. This value MUST be a **Realm** (section 2.2.1.34)

Flag: Representing the flags for this app principal. This value MUST be an **AppPrincipalFlags** (section 2.2.2.1)

2.2.4.6 App Principals with App Instance

The **App Principals with App Instance** result set contains metadata about an app instance and its app principal.

```
CompositePartitionKey varbinary(529),
Title nvarchar(255),
RedirectUrl nvarchar(255),
Realm nvarchar(255),
Flag int,
AppInstanceId uniqueidentifier,
DeploymentData nvarchar(max),
```

CompositePartitionKey: The value identifying the site subscription identifier, app principal identifier pair. This value MUST be an **AppRegistryCompositePartitionKey** (section 2.2.3.1).

Title: The title of the app of the app instance. This value MUST be an **AppName** (section 2.2.1.4).

RedirectUrl: The redirect URL of the app principal. This value MUST be a **RedirectUrl** (section 2.2.1.33).

Realm: The app principal realm of the app principal. This value MUST be a **Realm** (section 2.2.1.34).

Flag: The state flags of the app. This value MUST be an **AppPrincipalFlags** (section 2.2.2.1).

AppInstanceId: The app instance identifier. The value MUST be an **AppInstanceIdentifier** (section 2.2.1.36).

DeploymentData: The deployment data. The value MUST be a **DeploymentData** (section 2.2.1.27).

2.2.4.7 App Principal Permissions

The **App Principal Permissions** result set contains information about the app principal permissions.

```
ProviderTypeId uniqueidentifier,
Perm varbinary(max),
Version int,
```

ProviderTypeId: The unique identifier of an app principal permission provider. This value MUST be a **ProviderTypeId** (section 2.2.1.32).

Perm: Implementation specific representation of the app principal permission. This value MUST be a **Permission** (section 2.2.1.40).

Version: The current app principal permission version. This value MUST be a **Version** (section 2.2.1.41).

2.2.4.8 Apps

The **Apps** result set contains information about apps that are installed from the marketplace.

```
CompositePartitionKey varbinary(33),
AppName nvarchar(1024),
AssetId nvarchar(14),
```

CompositePartitionKey: The composite partition key identifying the site subscription identifier and the app product identifier. The value MUST be an **AppMonitoringCompositePartitionKey.** (section 2.2.3.4)

AppName: The title of the app. The value MUST be an **AppName** (section 2.2.1.4).

AssetId: The marketplace asset identifier of the app. The value MUST be an **AssetId** (section 2.2.1.3).

2.2.4.9 Licensed Apps

The **Licensed Apps** result set contains a list of apps which have at least one marketplace license in the protocol server.

```
CompositePartitionKey varbinary(33),
AppName nvarchar(1024),
AppDate datetime,
```

CompositePartitionKey: The composite partition key identifying the site subscription identifier and the app product identifier. The value MUST be an **AppLicensingCompositePartitionKey** (section 2.2.3.2).

AppName: The title of the app. The value MUST be an **AppName** (section 2.2.1.4).

AppDate: The appearance date of the app. The value MUST be an **AppDate** (section 2.2.1.10).

2.2.4.10 Manageable Apps

The **Manageable Apps** result set contains information about apps that a marketplace license director can manage at least one marketplace license of.

```
CompositePartitionKey varbinary(33),
AppName nvarchar(1024),
AppDate datetime,
LicenseType tinyint,
IsLicenseExpired bit,
IsTokenExpired bit,
```

CompositePartitionKey: The composite partition key identifying the site subscription identifier and the app product identifier. The value MUST be an **AppLicensingCompositePartitionKey** (section 2.2.3.2)

AppName: The title of the app for this marketplace license. The value MUST be an **AppName** (section 2.2.1.4).

AppDate: The date when the app was acquired. The value MUST be a **LicenseAcquisitionDate** (section 2.2.1.9).

LicenseType: The marketplace license type. The value MUST be a **LicenseType** (section 2.2.2.2).

IsLicenseExpired: Whether this marketplace license has expired. The value MUST be an **IsLicenseExpired** (section 2.2.1.13).

IsTokenExpired: Whether the token for this marketplace license has expired. The value MUST be an **IsTokenExpired** (section 2.2.1.14).

2.2.4.11 Item Count

The **Item Count** result set indicates the number of rows contained in the other result set returned by the stored procedure that returns the **Item Count** result set.

```
ItemCount int,
```

ItemCount: The number of items returned in the other result set returned by the stored procedure. This value MUST be an **ItemCount**(section 2.2.1.38).

2.2.4.12 License Directors

The **License Directors** result set contains information about marketplace license directors. Each row in the result set MUST contain all the attributes of a single marketplace license director.

```
LicenseDirectorIdentity nvarchar(255),
LicenseDirectorKey nvarchar(255),
```

LicenseDirectorIdentity: The name of the security principal (2) of a marketplace license director. The value MUST be a **LicenseDirectorIdentity**(section 2.2.1.25).

LicenseDirectorKey: The key of the security principal (2) of a marketplace license director. The value MUST be a **LicenseDirectorKey**(section 2.2.1.26).

2.2.4.13 License Users

The **License Users** result set contains information about users to whom marketplace licenses are assigned. Each row in the result set MUST contain all the attributes of a marketplace license user assignment.

```
UserIdentity nvarchar(255),
UserKey nvarchar(255),
```

UserIdentity: The name of the security principal (2) of a user. The value MUST be a **UserIdentity**(section 2.2.1.5).

UserKey: The key of the security principal (2) of a user. The value MUST be a **UserKey**(section 2.2.1.6). The value MUST NOT be NULL.

2.2.4.14 Licenses

Licenses result set contains information about marketplace licenses.

```
RawXMLEntitlementToken nvarchar(512),
ContentMarket nvarchar(10),
BillingMarket nvarchar(2),
CompositePartitionKey varbinary(33),
LicenseId uniqueidentifier,
LicenseType tinyint,
PurchaserIdentity nvarchar(16),
MaxUserCount int,
CurrentUserCount int,
ExpirationDate datetime,
AssetId nvarchar(14),
DeploymentId uniqueidentifier,
LicenseAcquisitionDate datetime,
TokenExpiryDate datetime,
IsTokenExpired bit,
IsLicenseExpired bit,
OmexLicenseType tinyint,
```

RawXMLEntitlementToken: Includes the **marketplace license token**. This value MUST be a **RawXMLEntitlementToken** (section 2.2.1.37).

ContentMarket: The marketplace content market of the marketplace license. This value MUST be a **ContentMarket** (section 2.2.1.28).

BillingMarket: The marketplace billing market of the marketplace license. This value MUST be a **BillingMarket** (section 2.2.1.29).

CompositePartitionKey: The composite partition key identifying the site subscription identifier of the site subscription, in the scope of which the marketplace license is valid and the app product identifier of the app for which the marketplace license is issued. The value MUST be an **AppLicensingCompositePartitionKey** (section 2.2.3.2).

LicenseId: The identifier of a marketplace license in the protocol server. This value MUST be a **LicenseId** (section 2.2.1.2).

LicenseType: The marketplace license type. The value MUST be a **LicenseType** (section 2.2.2.2).

PurchaserIdentity: The identifier of a user that purchased this marketplace license. This value must be a **PurchaserIdentity** (section 2.2.1.8).

MaxUserCount: The maximum number of marketplace license user assignments allowed for this marketplace license. This value MUST be -1 if **LicenseType** value is 1 or 3, and a positive integer if **LicenseType** value is 2 or 4.

CurrentUserCount: The current number of marketplace license user assignments for this marketplace license. This value MUST be NULL if **LicenseType** value is 1 or 3, and a non-negative integer if **LicenseType** value is 2 or 4.

ExpirationDate: The trial period end date of this marketplace license. The value MUST be an **ExpirationDate** (section 2.2.1.11). If **LicenseType** is 0 or 1, **ExpirationDate** MUST be null. If **LicenseType** is 2 or 3, **ExpirationDate** MUST NOT be null.

AssetId: The marketplace asset identifier of the app for which this marketplace license is issued. This value MUST be an **AssetId** (section 2.2.1.3).

DeploymentId: The marketplace deployment identifier to which this marketplace license is issued. This value MUST be a **DeploymentId** (section 2.2.1.17).

LicenseAcquisitionDate: Date and time of the issuing of the marketplace license. This value MUST be a **LicenseAcquisitionDate** (section 2.2.1.9).

TokenExpiryDate: Expiration date of the marketplace license token of the marketplace license. This value MUST be a **TokenExpiryDate** (section 2.2.1.12).

IsTokenExpired: The value indicating whether the marketplace license token of the marketplace license has expired or whether the automatic renewal readiness status of the marketplace license is preserved(section 3.2.1). This value MUST be an **IsTokenExpired** (section 2.2.1.14). This value MUST be 1 if the token of the marketplace license has expired or the automatic renewal readiness status of the marketplace license is preserved(section 3.2.1), and MUST be 0 otherwise.

IsLicenseExpired: The value indicating whether the trial period of a marketplace license has expired or not. This value MUST be an **IsLicenseExpired** (section 2.2.1.13). If the **LicenseType** is 0 or 1, this value MUST be 0.

OmexLicenseType: The commercial type of the marketplace license. This value MUST be an **OmexLicenseType** (section 2.2.2.3).

2.2.4.15 Manageable App Licenses

The **Manageable App Licenses** result set contains information about marketplace licenses that a marketplace license director can manage.

```
CompositePartitionKey varbinary(33),
LicenseId uniqueidentifier,
LicenseType tinyint,
PurchaserIdentity nvarchar(16),
PurchaserSPIdentity nvarchar(255),
MaxUserCount int,
CurrentUserCount int,
ExpirationDate datetime,
AppName nvarchar(1024),
RightsLevel tinyint,
OmexLicenseType tinyint,
```

CompositePartitionKey: The composite partition key identifying the site subscription identifier and the app identifier. The value MUST be an **AppLicensingCompositePartitionKey**(section 2.2.3.2).

LicenseId: The unique identifier of this marketplace license. The value MUST be a **LicenseId**(section 2.2.1.2).

LicenseType: The marketplace license type. The value MUST be a **LicenseType** (section 2.2.2.2).

PurchaserIdentity: The identifier of a user that purchased this marketplace license (section 2.2.1.8).

PurchaserSPIdentity: The name for the security principal (2) who is the purchaser of this marketplace license. The value MUST be a **PurchaserSPIdentity**(section 2.2.1.7).

MaxUserCount: The maximum number of marketplace license user assignments allowed for this marketplace license. This value MUST be -1 if **LicenseType** (section 2.2.2.2) value is 1 or 3, and a positive integer if **LicenseType** (section 2.2.2.2) value is 2 or 4. This value MUST be a **MaxUserCount** (section 2.2.1.15).

CurrentUserCount: The current number of marketplace license user assignments for this marketplace license. This value MUST be NULL if **LicenseType** (section 2.2.2.2) value is 1 or 3, and a non-negative integer if **LicenseType** (section 2.2.2.2) value is 2 or 4. This value MUST be a **CurrentUserCount** (section 2.2.1.16).

ExpirationDate: The expiration date of this marketplace license. This value MUST be null if **LicenseType** (section 2.2.2.2) is 0 or 1. This value MUST NOT be null if **LicenseType** (section 2.2.2.2) is 2 or 3. The value MUST be an **ExpirationDate** (section 2.2.1.11).

AppName: The title of the app for this marketplace license. The value MUST be an **AppName**(section 2.2.1.4).

IconUrl: The URL of the icon image of the app. This value MUST be an **IconUrl** (section 2.2.1.30).

ProviderName: The name of the publisher of the app. This value MUST be a **ProviderName** (section 2.2.1.31)

RightsLevel: The level of rights for the marketplace license director who is retrieving this marketplace license. The value MUST be a **RightsLevel**(section 2.2.2.4).

OmexLicenseType: The commercial type of the marketplace license. This value MUST be an **OmexLicenseType** (section 2.2.2.3).

2.2.4.16 Manageable App Identifiers

The Manageable App Identifiers result set contains a collection of composite partition keys that represent a combination of an app product identifier and a site subscription identifier.

```
CompositePartitionKey varbinary(33),
```

CompositePartitionKey: A composite partition key identifying the app product identifier and the site subscription identifier of the app.

2.2.5 Tables and Views

None.

2.2.6 XML Structures

No common XML structures are defined in this protocol.

2.2.6.1 Namespaces

This specification does not define any common **XML schema** namespaces.

2.2.6.2 Simple Types

This specification does not define any common XML schema simple type definitions.

2.2.6.3 Complex Types

This specification does not define any common XML schema complex type definitions.

2.2.6.4 Elements

This specification does not define any common XML schema element definitions.

2.2.6.5 Attributes

This specification does not define any common XML schema attribute definitions.

2.2.6.6 Groups

This specification does not define any common XML schema group definitions.

2.2.6.7 Attribute Groups

This specification does not define any common XML schema attribute group definitions.

2.2.7 User-Defined Table Types

This section defines common user-defined table types appearing as input parameter types in stored procedures that are used by this protocol specification.

2.2.7.1 User List

This table valued type defines a set of security principals (2).

UserIdentity: Name of the security principal (2).The value MUST be a **UserIdentity** (section 2.2.1.5).

UserKey: Key of the security principal (2). The value MUST be a **UserKey** (section 2.2.1.6). This value MUST NOT be NULL.

2.2.7.2 Composite Partition Key List

This table valued type defines a set of composite partition keys. All composite partition keys in a **Composite Partition Key List** represent one of the following:

- **App Principal:** A site subscription identifier with an app principal identifier
- **App:** A site subscription identifier with an app product identifier.
- **Site Subscription Identifier:** A single site subscription identifier.

CompositePartitionKey: The composite partition key, which MUST be either an **AppRegistryCompositePartitionKey** (section 2.2.3.1) if each element is representing an app principal, an **AppLicensingCompositePartitionKey** (section 2.2.3.2) if each element is representing an app or a **DeploymentIdCompositePartitionKey** (section 2.2.3.3) if each element is representing a site subscription identifier.

2.2.7.3 Unique License Identifiers List

This table valued type defines a set of data tuples, each of which uniquely identifies a marketplace license stored in the protocol server.

CompositePartitionKey: The composite partition key identifying the app product identifier and the site subscription identifier for the app of the marketplace license.This value MUST be an **AppLicensingCompositePartitionKey** (section 2.2.1.2).

PurchaserIdentity: The identifier for the purchaser of the marketplace license. This value MUST be a **PurchaserIdentity** (section 2.2.1.8)

2.2.7.4 License List

This table valued type defines update information for a set of marketplace licenses.

CompositePartitionKey: The composite partition key identifying the app product identifier and the site subscription identifier for the app of the marketplace license. This value MUST be an **AppLicensingCompositePartitionKey** (section 2.2.1.2).

LicenseId: The GUID uniquely identifying a marketplace license stored in the protocol server. This value MUST be a **LicenseId** (section 2.2.1.2).

LicenseType: The marketplace license type. This value MUST be a **LicenseType** (section 2.2.2.2).

OmexLicenseType: The commercial type of the marketplace license. This value MUST be an **OmexLicenseType** (section 2.2.2.3).

PurchaserIdentity: The identifier for the purchaser of the marketplace license. This value MUST be a **PurchaserIdentity** (section 2.2.1.8).

MaxUserCount: The maximum number of marketplace license user assignments allowed for the marketplace license. This value MUST be a **MaxUserCount** (section 2.2.1.15). This value MUST be -1 if **LicenseType** value is 1 or 3, and a positive integer if **LicenseType** value is 2 or 4.

ExpirationDate: Expiration date and time of the marketplace license. This value MUST be an **ExpirationDate** (section 2.2.1.11). If **LicenseType** is 0 or 1, **ExpirationDate** MUST be NULL. If **LicenseType** is 2 or 3, **ExpirationDate** MUST NOT be NULL.

AssetId: Marketplace asset identifier for the app of the marketplace license. This value MUST be an **AssetId** (section 2.2.1.3).

DeploymentId: Marketplace deployment identifier to which the marketplace license is tied. This value MUST be a **DeploymentId** (section 2.2.1.17).

LicenseAcquisitionDate: Date and time when the marketplace license was issued. This value MUST be a **LicenseAcquisitionDate** (section 2.2.1.9).

TokenExpiryDate: Expiration date for the marketplace license token of the marketplace license. This value MUST be a **TokenExpiryDate** (section 2.2.1.12).

ContentMarket: Marketplace content market of the marketplace license. This value MUST be a **ContentMarket** (section 2.2.1.28).

BillingMarket: Marketplace billing market of the marketplace license. This value MUST be a **BillingMarket** (section 2.2.1.29).

RawXMLEntitlementToken: Marketplace license token of the marketplace license. The value MUST be a **RawXMLEntitlementToken** (section 2.2.1.37).

3 Protocol Details

3.1 Common Details

None.

3.2 Server Details

The back-end database protocol server responds only to stored procedure calls from the protocol client. It returns result sets and return codes and never initiates communication with other endpoints of the protocol.

3.2.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

On an abstract level, the protocol server stores the following entities:

- app principals
- app principal permissions
- Apps (not the apps themselves, but information about them)
- marketplace licenses
- marketplace deployment identifiers
- marketplace license user assignments
- marketplace license director**s**
- site subscriptions
- app instance**s**

A site subscription has the following attributes:

- One site subscription identifier.

A site subscription is associated with:

- Zero or more app principals
- Zero or more app principal permissions.
- Zero or more marketplace licenses.
- Zero or one marketplace deployment identifier.

An app principal entity has the following attributes:

- One composite partition key which identifies the app principal identifier and the site subscription identifier of the app principal.
- One **app principal identifier**, which MUST be a UTF-16 string of at most 256 characters.
- One title.
- Zero or one redirect URL.
- Zero or one realm.
- One status flag.

Every app principal has an **app principal status**, which can be in zero or more of the following states:

- **Disabled:** The app principal is disabled and cannot perform any function.
- **HasAnonymousAccess:** The app principal can, if given rights, access resources on its own without using the rights of a user.

An app principal is associated with:

- Zero or one app instance
- Zero or more site subscription/app pairs. Whenever there is such an association, we alternatively say that the app principal is *registered* with the app in the scope of the site subscription. Such an association itself is associated with one unique app instance identifier.
- Zero or more app principal permissions.
- One site subscription.

An app principal permission entity has the following attributes:

- One permission provider.
- One version.
- One permissions set, which includes the complete set of permissions assigned to the app principal related to the app principal permissions.

An app principal permission entity is associated with:

- One app principal.

An app entity has the following attributes:

- One composite partition key which represents the app product identifier and site subscription identifier of the app.
- One app product identifier, which is unique to the app.
- One title.
- Zero or one icon image location.
- One provider name.

- One marketplace asset identifier.

An app entity is associated with:

- Zero or more marketplace licenses.
- Zero or more site subscription/app principal pairs, when the app is registered with the app principal in the scope of the site subscription.

A marketplace license entity has the following attributes:

- One identifier.
- One license type.
- One commercial type.
- Zero or one limit on the number of users that can be assigned to it. The count is zero when the marketplace license type of the marketplace license is *PerpetualAllUsers* and *TrialAllUsers*. The count is one when the marketplace license type of the marketplace license is *PerpetualMultiUser* and *TrialMultiUser*.
- Zero or one expiration date. The count is zero when the marketplace license type of the marketplace license is *PerpetualAllUsers* and *PerpetualMultiUser*. The count is one when the marketplace license type of the marketplace license is *TrialAllUsers* and *TrialMultiUser*.
- One acquisition date.
- One token expiry date.
- One marketplace license token.
- One marketplace license type.
- One purchaser name, which is the name of the security principal (2) of the user representing the purchaser of the marketplace license.
- One purchaser key, which is the key of the security principal (2) of the user representing the purchaser of the marketplace license. Among the marketplace licenses associated with the same site subscription/app pair, this key MUST be unique.
- One marketplace content market
- One marketplace billing market

The transaction approval status of a marketplace license indicates whether purchasing transaction of the marketplace license has been approved or not. The transaction approval status has the following states:

- **Pending:** Indicates that the purchasing transaction of the marketplace license has not been approved yet.
- **Non-pending:** Indicates that the purchasing transaction of the marketplace license has been approved.

Every marketplace license stored in the protocol server MUST be in one of these states in order for the following stored procedures to function correctly:

- **proc_AM_ImportLicense** (see section 3.2.5.21)
- **proc_AM_BulkImportLicense** (see section 3.2.5.6)

The automatic renewal readiness status of a marketplace license at a specific time (as measured by the clock of the computer hosting the protocol server) indicates whether the marketplace license qualifies for an automatic renewal attempt at that time or not. Note that this status is not static and subject to change in time. The automatic renewal readiness status has the following states:

- **Ready:** The marketplace license qualifies for an automatic renewal attempt.
- **Not-Ready:** The marketplace license does not, but will qualify for an automatic renewal attempt.
- **Preserved:** The marketplace license does not and will never qualify for an automatic renewal attempt, unless its automatic renewal readiness status is explicitly set to **Ready** by one of the stored procedures included in the protocol.

Every marketplace license stored in the protocol server MUST be in one of these states in order for the following stored procedures to function correctly:

- **proc_AM_ImportLicense** (see section 3.2.5.21)
- **proc_AM_BulkImportLicense** (see section 3.2.5.6)
- **proc_AM_AcquireEntitlementsForRenewal** (see section 3.2.5.1)
- **proc_AM_GetManageableApps** (see section 3.2.5.19)
- **proc_AM_CheckLicense** (see section 3.2.5.7)

A marketplace license is associated with:

- One site subscription.
- One app.
- One marketplace deployment identifier.
- Zero or more marketplace license user assignments.
- Zero or more marketplace license directors.
- One marketplace content market.
- One marketplace billing market.

A marketplace deployment identifier is associated with:

- One site subscription.
- Zero or more marketplace licenses.

A marketplace license user assignment has the following attributes:

- One user identity, which is the name of the security principal (2) representing the user assigned.
- One user key, which is the key of the security principal (2) representing the user assigned.
- One assignment date/time.

A marketplace license user assignment is associated with:

- One marketplace license. The user key MUST be unique among the marketplace license user assignments associated with the same marketplace license.

A marketplace license director has the following attributes:

- One license director identity, which is the name of the security principal (2) representing the marketplace license director.
- One license director key, which is the key of the security principal (2) representing the marketplace license director.
- One set of license director rights.

A marketplace license director is associated with:

- One marketplace license. The license director key MUST be unique among the marketplace license directors associated with the same marketplace license.

An app instance entity has the following attributes:

- One app instance identifier.
- One deployment data set.

3.2.2 Timers

None.

3.2.3 Initialization

A connection that uses the underlying protocol layers that are specified in section 1.4 MUST be established before using this protocol, as specified in [MS-TDS].

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

3.2.5.1 proc_AM_AcquireEntitlementsForRenewal

The **proc_AM_AcquireEntitlementsForRenewal** **stored procedure** is called to get the first specified number of marketplace licenses with automatic renewal readiness status ready (section 3.2.1), ordered ascending first by the composite partition key and then by the marketplace license identifier, having their composite partition keys and marketplace license identifiers come strictly (exclusively) after the *@RangeStart*/*@StartingLicenseId* composite partition key/marketplace license identifier pair based on the same ordering scheme.

Upon return from this stored procedure, the *@RangeStart* and *@RangeEnd* parameters MUST be set according to the following:

· If *@RangeStart* is greater than the end point of the protocol server data range, then *@RangeStart* and *@RangeEnd* MUST be set to the protocol server data range end point.

· If *@RangeEnd* is less than the start point of the protocol server data range, then *@RangeStart* and *@RangeEnd* MUST be set to the protocol server data range start point.

· If *@RangeStart* and *@RangeEnd* intersect with the protocol server data range, then *@RangeStart* MUST be set to the maximum of the *@RangeStart* value and protocol server data range start point, and *@RangeEnd* MUST be set to the minimum of the *@RangeEnd* value and the protocol server data range end point.

· NULL is the maximum value for any data range point comparison.

```
PROCEDURE proc_AM_AcquireEntitlementsForRenewal (
@RangeStart varbinary(529) OUTPUT
,@StartingLicenseId uniqueidentifier
,@BatchSize int
,@RangeEnd varbinary(529) OUTPUT
);
```

@RangeStart: The binary value which, along with the *@StartingLicenseId*, determines the lower limit on the ascending order of composite partition key/marketplace license identifier tuples that the stored procedure will use in evaluating the result set. This value MUST be an **AppLicensingCompositePartitionKey** (section 2.2.3.2).

@StartingLicenseId: The value which, along with the *@RangeStart*, determines the lower limit on the ascending order of composite partition key/marketplace license identifier tuples that the stored procedure will use in evaluating the result set. This value MUST be a **LicenseId** (section 2.2.1.2).

@BatchSize: The maximum number of marketplace licenses to be returned. This value MUST be an **LimitCount**(section 2.2.1.20).

@RangeEnd: The binary value which determines the composite partition key part of the upper limit on the ascending order of composite partition key/marketplace license identifier tuples that the stored procedure will use in evaluating the result set. Marketplace license identifier part of the upper limit is the GUID consisting of all zeros. This value MUST be an **AppLicensingCompositePartitionKey** (section 2.2.3.2).

Return Values: MUST NOT return any values.

Result Sets:

Licenses

This stored procedure MUST return a Licenses.

3.2.5.2 proc_AM_AddApp

The **proc_AM_AddApp** stored procedure is called to add an app for the purpose of runtime monitoring. If an app with the specified app product identifier and the site subscription identifier already exists, the existing app is updated with the specified values.

```
PROCEDURE proc_AM_AddApp (
@CompositePartitionKey binary(33)
,@AppName nvarchar(1024)
,@AssetId nvarchar(14)
,@ErrorCode int OUTPUT
);
```

@CompositePartitionKey: The composite partition key identifying the app product identifier and the site subscription identifier of the app. The value MUST be an **AppMonitoringCompositePartitionKey** (section 2.2.3.4).

@AppName: The title of the app. The value MUST be an **AppName** (section 2.2.1.4).

@AssetId: The marketplace asset identifier of the app. The value MUST be an **AssetId** (section 2.2.1.3).

@ErrorCode: This is the error code representing success or failure of the add operation. This takes values from the following table.

Value	Description
-1000	The *@CompositePartitionKey* value is outside the data range of the protocol server.
0	No errors encountered.

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST NOT return any result sets.

3.2.5.3 proc_AM_AddLicenseDirectors

The **proc_AM_AddLicenseDirectors** stored procedure is called to add a set of marketplace license directors to a marketplace license. If a marketplace license director already exists in the given marketplace license it MUST be ignored and not added again.

```
PROCEDURE proc_AM_AddLicenseDirectors (
@Users tvpUserList
,@LicenseId uniqueidentifier
,@CompositePartitionKey varbinary(33)
,@UserKey nvarchar(255)
,@ErrorCode int OUTPUT
);
```

@Users: The list of users that will be added as a marketplace license director to the given marketplace license. The value MUST be a **User List** (section 2.2.7.1).

@LicenseId: The identifier representing a marketplace license. The value MUST be a **LicenseId** (section 2.2.1.2).

@CompositePartitionKey: The composite partition key identifying the app product identifier and the site subscription identifier for the app of the marketplace license to add the marketplace license directors to. The value MUST be an **AppLicensingCompositePartitionKey** (section 2.2.3.2)

@UserKey: The key of the marketplace license director adding the marketplace license directors. If a SharePoint Farm Administrator is doing the request, this value MUST be NULL. The value MUST be a **UserKey**(section 2.2.1.6).

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

Value	Description
-1000	The *@CompositePartitionKey* value is outside the data range of the protocol server.
-17	The number of marketplace license directors has exceed the maximum allowed. The maximum allowed MUST be 100.
-14	The marketplace license type is neither *PerpetualMultiUser* nor *TrialMultiUser*.
-8	The specified marketplace license does not exist.
-4	The specified user is not a marketplace license director of the specified marketplace license.
0	No errors encountered.

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return a License Directors.

3.2.5.4 proc_AM_AddUserLicenseAssignments

The **proc_AM_AddUserLicenseAssignments** stored procedure is called to assign a marketplace license to a set of users on behalf of a marketplace license director or SharePoint Farm Administrator. The stored procedure returns the set of specified users who already have the specified marketplace license assigned to them.

```
PROCEDURE proc_AM_AddUserLicenseAssignments (
@Users tvpUserList
,@LicenseId uniqueidentifier
,@CompositePartitionKey varbinary(33)
,@UserKey nvarchar(255)
,@ErrorCode int OUTPUT
);
```

@Users: The set of users. This value MUST be a **User List** (section 2.2.7.1).

@LicenseId: The identifier of the marketplace license.

@CompositePartitionKey: The composite partition key that specifies the app product identifier and the site subscription identifier. The value MUST be an **AppLicensingCompositePartitionKey** (section 2.2.3.2).

@UserKey: The key of the marketplace license director adding the marketplace license user assignments. If a SharePoint Farm Administrator is doing the request, this value MUST be NULL. The value MUST be a **UserKey**(section 2.2.1.6).

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

Value	Description
-1000	The @*CompositePartitionKey* value is outside the data range of the protocol server.

Value	Description
-14	The marketplace license type is neither *PerpetualMultiUser* nor *TrialMultiUser*.
-8	The specified marketplace license does not exist.
-6	The assignment fo the specified marketplace license to the specified users would exceed the maximum number of available marketplace license user assignments of the marketplace license.
-4	The specified user is not a marketplace license director of the specified marketplace license.
0	No errors encountered.

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return a License Users.

3.2.5.5 proc_AM_BulkGetDisplayableAppInfo

The **proc_AM_BulkGetDisplayableAppInfo** is called to retrieve metadata about a set of apps. The composite partition key of the apps returned from this stored procedure MUST be inside the specified data range.

Upon return from this stored procedure, the *@RangeStart* and *@RangeEnd* parameters MUST be set according to the following:

· If *@RangeStart* is greater than the end point of the protocol server data range, then *@RangeStart* and *@RangeEnd* MUST be set to the protocol server data range end point.

· If *@RangeEnd* is less than the start point of the protocol server data range, then *@RangeStart* and *@RangeEnd* MUST be set to the protocol server data range start point.

· If *@RangeStart* and *@RangeEnd* intersect with the protocol server data range, then *@RangeStart* MUST be set to the maximum of the *@RangeStart* value and protocol server data range start point, and *@RangeEnd* MUST be set to the minimum of the *@RangeEnd* value and the protocol server data range end point.

· NULL is the maximum value for any data range point comparison.

```
PROCEDURE proc_AM_BulkGetDisplayableAppInfo (
@CompositePartitionKeyList tvpCompositePartitionKeyList
,@RangeStart varbinary(529) OUTPUT
,@RangeEnd varbinary(529) OUTPUT
);
```

@CompositePartitionKeyList: The list containing the app product identifiers and site subscription identifiers of the apps. The value MUST be a **Composite Partition Key List (App)** (section 2.2.7.2).

@RangeStart: The beginning of the data range in which the results will be looked for. The value MUST be a **RangeStart** (section 2.2.1.22).

@RangeEnd: The end of the data range in which the results will be looked for. The value MUST be a **RangeEnd** (section 2.2.1.23).

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return a Displayable App Information.

3.2.5.6 proc_AM_BulkImportLicense

The **proc_AM_BulkImportLicense** is called to do a bulk import of marketplace licenses during a marketplace license renewal process. All apps of the marketplace licenses processed with no errors by this stored procedure MUST have a composite partition key inside the specified data range. It MUST process the marketplace licenses in the following way:

1. *@RenewedEntitlements*: These marketplace licenses MUST be updated with the passed in data. For any particular marketplace license, if the token expiry date is sooner than 10 days, as measured by the clock of the computer hosting the protocol server, the marketplace license is marked as pending. In this case the marketplace license will be qualified for automatic renewal attempt 5 hours before token expiry date. If token expiry date is later than 10 days or more, as measured by the clock of the computer hosting the protocol server, the marketplace license is marked as non-pending. In this case the marketplace license will be qualified for automatic renewal attempt 72 hours before token expiry date.

2. *@EntitlementsToBeDeleted:* These marketplace licenses MUST be deleted along with the marketplace license directors and the marketplace license user assignments associated to these marketplace licenses.

3. *@EntitlementsToBePreserved:* These marketplace licenses MUST be marked preserved so that they will no longer qualify for automatic renewal.

4. *@EntitlementsToBeRetried:* Each marketplace license in this list MUST be marked in such a way that automatic renewal is attempted again in 6 hours when the marketplace license is marked as non-pending or 1 minute when the marketplace license is marked as pending.

Upon return from this stored procedure, the *@RangeStart* and *@RangeEnd* parameters MUST be set according to the following:

· If *@RangeStart* is greater than the end point of the protocol server data range, then *@RangeStart* and *@RangeEnd* MUST be set to the protocol server data range end point.

· If *@RangeEnd* is less than the start point of the protocol server data range, then *@RangeStart* and *@RangeEnd* MUST be set to the protocol server data range start point.

· If *@RangeStart* and *@RangeEnd* intersect with the protocol server data range, then *@RangeStart* MUST be set to the maximum of the *@RangeStart* value and protocol server data range start point, and *@RangeEnd* MUST be set to the minimum of the *@RangeEnd* value and the protocol server data range end point.

· NULL is the maximum value for any data range point comparison.

The procedure MUST return the list of marketplace licenses that got updated and the result of each individual operation.

```
PROCEDURE proc_AM_BulkImportLicense (
@RenewedEntitlements tvpLicenseList
,@EntitlementsToBeDeleted tvpUniqueLicenseIdentifiersList
,@EntitlementsToBePreserved tvpUniqueLicenseIdentifiersList
,@EntitlementsToBeRetried tvpUniqueLicenseIdentifiersList
,@RangeStart varbinary(529) OUTPUT
,@RangeEnd varbinary(529) OUTPUT
);
```

@RenewedEntitlements: The list of marketplace licenses to be updated. The value MUST be a **License List** (section 2.2.7.4).

@EntitlementsToBeDeleted: The list of marketplace licenses to be deleted. The value MUST be a **Unique License Identifiers List** (section 2.2.7.3).

@EntitlementsToBePreserved: The list of marketplace licenses to be marked as preserved so that they no longer qualify for the automatic renewal process. The value MUST be a **Unique License Identifiers List** (section 2.2.7.3).

@EntitlementsToBeRetried: The list of marketplace licenses to be marked such that they will qualify for the automatic renewal process sooner. The value MUST be a **Unique License Identifiers List** (section 2.2.7.3).

@RangeStart: The beginning of the data range to operate on. The value MUST be a **RangeStart** (section 2.2.1.22).

@RangeEnd: The end of the data range to operate on. The value MUST be a **RangeEnd** (section 2.2.1.23).

Return Values: An integer which MUST be 0.

Result Sets:

this stored procedure MUST return a License Import Results

3.2.5.7 proc_AM_CheckLicense

The **proc_AM_CheckLicense** stored procedure is called to get all marketplace licenses satisfying all of the following conditions:

- The marketplace license belongs to the app having the app product identifier encoded in *@CompositePartitionKey* (for details of the encoding, see section 2.2.1.2).
- The marketplace license is valid in the site subscription having the site subscription identifier encoded in *@CompositePartitionKey* (for details of the encoding, see section 2.2.1.2).
- The marketplace license satisfies one of the following:
  - It is of type PerpetualAllUsers.
  - It is of type TrialAllUsers.
  - It is of type PerpetualMultiUser or TrialMultiUser, and it has a marketplace license user assignment for the security principal (2) identified by the *@UserKey*.

If *@RequireAppAuthentication* has the value 1, the results are returned only if the app product identifier encoded in *@CompositePartitionKey* is registered with the app principal identifier *@AppId*,

in the scope of the site subscription with the site subscription identifier encoded in *@CompositePartitionKey* (see section 3.2.1 for the concept of registration).

```
PROCEDURE proc_AM_CheckLicense (
@UserKey nvarchar(255)
,@CompositePartitionKey varbinary(33)
,@AppId nvarchar(256)
,@RequireAppAuthentication bit
,@ErrorCode int OUTPUT
);
```

@UserKey: The key of the current user security principal (2). This value MUST be a **UserKey** (see section 2.2.1.6)

@CompositePartitionKey: The composite partition key that determines the site subscription identifier and the app product identifier to get the marketplace licenses for. The value MUST be an **AppLicensingCompositePartitionKey** (section 2.2.3.2).

@AppId: The App Principal Identifier. The value MUST be an **AppId** (section 2.2.1.18).

@RequireAppAuthentication: Specifies whether to check if the app principal identifier *@AppId* is registered with the app product identifier encoded in the *@CompositePartitionKey* (section 2.2.1.2 for encoding details). The value MUST be either 0 or 1.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

Value	Description
-1000	The *@CompositePartitionKey* value is outside the data range of the protocol server.
-11	The app principal identifier *@AppId* is not registered with the app product identifier encoded in *@CompositePartitionKey*.
0	No errors encountered.

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return a Licenses.

3.2.5.8 proc_AM_ClearAppInstanceDeploymentData

The **proc_AM_ClearAppInstanceDeploymentData** stored procedure is called to clear the app instance identifier of a specified app principal, and the deployment data of that app instance. This stored procedure MUST NOT have any effect if the current app instance identifier of the specified app principal is not the specified app instance identifier.

```
PROCEDURE proc_AM_ClearAppInstanceDeploymentData (
@CompositePartitionKey varbinary(529)
,@AppInstanceId uniqueidentifier
,@RequestGuid uniqueidentifier = null OUTPUT
);
```

@CompositePartitionKey: The composite partition key identifying the app principal identifier and the site subscription identifier. This value MUST be an **AppRegistryCompositePartitionKey** (section 2.2.3.1).

@AppInstanceId: The app instance identifier. The value MUST be an **AppInstanceIdentifier** (section 2.2.1.36).

@RequestGuid: A GUID uniquely identifying the current request. The value MUST be a **RequestGuid** (section 2.2.1.1).<1>

Return Values: An integer from the following table:

Value	Description
-1000	The *@CompositePartitionKey* value is outside the data range of the protocol server.
0	No errors encountered.

Result Sets:

This stored procedure MUST NOT return any result sets.

3.2.5.9 proc_AM_GetAppPrincipal

The proc_AM_GetAppPrincipal stored procedure is called to retrieve an app principal with the specified app principal identifier and the site subscription identifier.

```
PROCEDURE proc_AM_GetAppPrincipal (
@CompositePartitionKey varbinary(529)
,@RequestGuid uniqueidentifier = null OUTPUT
);
```

@CompositePartitionKey: The composite partition key identifying the app principal identifier and the site subscription identifier of the app principal. This value MUST be an **AppRegistryCompositePartitionKey** (section 2.2.3.1).

@RequestGuid: A GUID uniquely identifying the current request. The value MUST be a **RequestGuid** (section 2.2.1.1). <2>

Return Values: An integer which MUST be 0.

Result Sets:

If the *@CompositePartitionKey* value is outside the data range of the protocol server, this stored procedure MUST return a Mapping Error.

Otherwise, this stored procedure MUST return an App Principals.

3.2.5.10 proc_AM_GetAppPrincipalPerms

The **proc_AM_GetAppPrincipalPerms** stored procedure is called to get the set of app principal permissions for a given app principal and app principal permission provider. If the value of *@ProviderTypeId* is NULL, the app principal permissions for all app principal permission providers are returned.

```
PROCEDURE proc_AM_GetAppPrincipalPerms (
@CompositePartitionKey varbinary(529)
,@ProviderTypeId uniqueidentifier
,@RequestGuid uniqueidentifier = null OUTPUT
);
```

@CompositePartitionKey: The composite partition key identifying the app principal identifier and site subscription identifier of the app principal to get the app principal permissions of. The value MUST be an **AppRegistryCompositePartitionKey** (see section 2.2.1.1).

@ProviderTypeId: The unique identifier of an app principal permission provider. This value MUST be a **ProviderTypeId** (section 2.2.1.32)

@RequestGuid: A GUID uniquely identifying the current request. The value MUST be a **RequestGuid** (section 2.2.1.1). <3>

Return Values: An integer which MUST be 0.

Result Sets:

If the *@CompositePartitionKey* value is outside the data range of the protocol server, this stored procedure MUST return a Mapping Error.

Otherwise, this stored procedure MUST return the following:

- an App Principals with App Instance.
- an App Principal Permissions.

3.2.5.11 proc_AM_GetApps

The **proc_AM_GetApps** stored procedure is called to get the details of a specified number of apps with composite partition keys inside the specified data range. If *@SearchString* is not NULL, each returned app MUST have a title that contains the *@SearchString* value.

Upon return from this stored procedure, the *@RangeStart* and *@RangeEnd* parameters MUST be set according to the following:

· If *@RangeStart* is greater than the end point of the protocol server data range, then *@RangeStart* and *@RangeEnd* MUST be set to the protocol server data range end point.

· If *@RangeEnd* is less than the start point of the protocol server data range, then *@RangeStart* and *@RangeEnd* MUST be set to the protocol server data range start point.

· If *@RangeStart* and *@RangeEnd* intersect with the protocol server data range, then *@RangeStart* MUST be set to the maximum of the *@RangeStart* value and protocol server data range start point, and *@RangeEnd* MUST be set to the minimum of the *@RangeEnd* value and the protocol server data range end point.

· NULL is the maximum value for any data range point comparison.

```
PROCEDURE proc_AM_GetApps (
@SearchString nvarchar(1024) = null
,@MaxRows int
```

```
,@RangeStart varbinary(529) OUTPUT
,@RangeEnd varbinary(529) OUTPUT
);
```

@SearchString: The string that specifies a sub-string for the titles of the apps to be returned. The value MUST be a **SearchString**(section 2.2.1.24).

@MaxRows: The maximum number of apps to be returned. The value MUST be a **LimitCount** (section 2.2.1.20).

@RangeStart: The beginning of the data range in which the results will be looked for. The value MUST be a **RangeStart** (section 2.2.1.22).

@RangeEnd: The end of the data range in which the results will be looked for. The value MUST be a **RangeEnd** (section 2.2.1.23).

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return an Apps.

3.2.5.12 proc_AM_GetDeploymentId

The **proc_AM_GetDeploymentId** stored procedure is called to get the marketplace deployment identifier associated with the site subscription identifier encoded in the @CompositePartitionKey (see section 2.2.1.3 for details of the encoding). If there is no marketplace deployment identifier associated with the given site subscription identifier, it randomly generates one and stores it in the protocol server.

```
PROCEDURE proc_AM_GetDeploymentId (
@CompositePartitionKey binary(17)
,@ErrorCode int OUTPUT
);
```

@CompositePartitionKey: Determines the site subscription identifier to get the marketplace deployment identifier for. The value MUST be a **DeploymentIdCompositePartitionKey** (section 2.2.3.3).

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

Value	Description
-1000	The *@CompositePartitionKey* value is outside the data range of the protocol server.
0	No errors encountered.

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return a Deployment ID.

3.2.5.13 proc_AM_GetEntitledApps

The **proc_AM_GetEntitledApps** stored procedure is called to get a specified number of apps that have at least one marketplace license satisfying the following conditions:

1. Either the marketplace license type is *PerpetualAllUsers* or *TrialAllUsers*, or the marketplace license type is *PerpetualMultiUser* or *TrialMultiUser* and it has a marketplace license user assignment for the security principal (2) identified by the *@UserKey*.
2. The token expiry date is greater than or equal to the current time, as measured by the clock of the computer hosting the protocol server.
3. If the marketplace license type is *TrialMultiUser* or *TrialAllUsers*, the expiration date for the marketplace license is greater than or equal to the current time, as measured by the clock of the computer hosting the protocol server.
4. The composite partition key is falling into the specified data range.

This MUST be the set of apps that the given user has been assigned a marketplace license for. This stored procedure returns an app acquisition date which MUST be the earliest of the date when any marketplace license with the marketplace license type *PerpetualAllUsers* or *TrialAllUsers* for the app was acquired and the date when the given user was assigned a marketplace license of the app with the marketplace license type is *PerpetualMultiUser* or *TrialMultiUser*.

Upon return from this stored procedure, the *@RangeStart* and *@RangeEnd* parameters MUST be set according to the following:

- If *@RangeStart* is greater than the end point of the protocol server data range, then *@RangeStart* and *@RangeEnd* MUST be set to the protocol server data range end point.
- If *@RangeEnd* is less than the start point of the protocol server data range, then *@RangeStart* and *@RangeEnd* MUST be set to the protocol server data range start point.
- If *@RangeStart* and *@RangeEnd* intersect with the protocol server data range, then *@RangeStart* MUST be set to the maximum of the *@RangeStart* value and protocol server data range start point, and *@RangeEnd* MUST be set to the minimum of the *@RangeEnd* value and the protocol server data range end point.
- NULL is the maximum value for any data range point comparison.

```
PROCEDURE proc_AM_GetEntitledApps (
@UserKey nvarchar(255)
,@MaxItemCountToFetch int
,@RangeStart varbinary(529) OUTPUT
,@RangeEnd varbinary(529) OUTPUT
);
```

@UserKey: The key of the current user security principal (2). The value MUST be a **UserKey** (section 2.2.1.6).

@MaxItemCountToFetch: The maximum number of apps to be returned. The value MUST be a **LimitCount** (section 2.2.1.20).

@RangeStart: The beginning of the data range in which the results will be looked for. The value MUST be a **RangeStart** (section 2.2.1.22).

@RangeEnd: The end of the data range in which the results will be looked for. The value MUST be a **RangeEnd** (section 2.2.1.23).

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return a Licensed Apps.

3.2.5.14 proc_AM_GetLicenseDirectorsForLicenseById

The **proc_AM_GetLicenseDirectorsForLicenseById** stored procedure is called to get all the marketplace license directors that a given marketplace license has.

```
PROCEDURE proc_AM_GetLicenseDirectorsForLicenseById (
@LicenseId uniqueidentifier
,@CompositePartitionKey varbinary(33)
,@UserKey nvarchar(255)
,@PageSize int
,@StartingUserNumber int
,@ErrorCode int OUTPUT
);
```

@LicenseId: The identifier representing a marketplace license. The value MUST be a **LicenseId** (section 2.2.1.2).

@CompositePartitionKey: The composite partition key identifying the app product identifier and the site subscription identifier for the app of the marketplace license to get the marketplace license directors from. The value MUST be an **AppLicensingCompositePartitionKey** (section 2.2.3.2).

@UserKey: The key of the marketplace license director requesting the marketplace license directors. If a SharePoint Farm Administrator is doing the request, this value MUST be NULL. The value MUST be a **UserKey**(section 2.2.1.6).

@PageSize: The maximum number of marketplace license directors to be returned. The value MUST be a **PageSize** (section 2.2.1.21).

@StartingUserNumber: The index of the first marketplace license director to be returned among the whole list of qualifying marketplace license directors based on an implementation specific ordering. The value MUST be a **StartingRowNumber** (section 2.2.1.19).

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

Value	Description
-1000	The *@CompositePartitionKey* value is outside the data range of the protocol server.
-8	The specified marketplace license does not exist.
-4	The specified user is not a marketplace license director of the specified marketplace license.
0	No errors encountered.

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return a License Directors.

This stored procedure MUST return an Item Count.

3.2.5.15 proc_AM_GetManageableAppLicenseById

The **proc_AM_GetManageableAppLicenseById** stored procedure is called to get the marketplace license which has the specified identifier and associated with an app that has the specified composite partition key. If the *@UserKey* value is not NULL and the specified marketplace license does not have a marketplace license director with the given key, an empty result set MUST be returned.

```
PROCEDURE proc_AM_GetManageableAppLicenseById (
@LicenseId uniqueidentifier
,@CompositePartitionKey varbinary(33)
,@UserKey nvarchar(255)
,@ErrorCode int OUTPUT
);
```

@LicenseId: The identifier representing a marketplace license. The value MUST be a **LicenseId** (section 2.2.1.2).

@CompositePartitionKey: The composite partition key identifying the app product identifier and the site subscription identifier of the app to get a marketplace license for. The value MUST be an **AppLicensingCompositePartitionKey** (section 2.2.3.2).

@UserKey: The key of the marketplace license director requesting the marketplace license. If a SharePoint Farm Administrator is doing the request, this value MUST be NULL. The value MUST be a **UserKey**(section 2.2.1.6).

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

Value	Description
-1000	The *@CompositePartitionKey* value is outside the data range of the protocol server.
-8	The specified marketplace license cannot be found.
0	No errors encountered.

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return a Manageable App Licenses.

3.2.5.16 proc_AM_GetManageableAppLicenses

The **proc_AM_GetManageableAppLicenses** stored procedure is called to get a specified number of marketplace licenses with composite partition keys inside the specified data range. If the *@UserKey* is not NULL, the returned marketplace licenses MUST have a marketplace license director with the given key. If the *@SearchString* is not NULL, each app associated with the returned marketplace licenses MUST have a title that contains the *@SearchString* value.

Upon return from this stored procedure, the *@RangeStart* and *@RangeEnd* parameters MUST be set according to the following:

· If *@RangeStart* is greater than the end point of the protocol server data range, then *@RangeStart* and *@RangeEnd* MUST be set to the protocol server data range end point.

· If *@RangeEnd* is less than the start point of the protocol server data range, then *@RangeStart* and *@RangeEnd* MUST be set to the protocol server data range start point.

· If *@RangeStart* and *@RangeEnd* intersect with the protocol server data range, then *@RangeStart* MUST be set to the maximum of the *@RangeStart* value and protocol server data range start point, and *@RangeEnd* MUST be set to the minimum of the *@RangeEnd* value and the protocol server data range end point.

· NULL is the maximum value for any data range point comparison.

```
PROCEDURE proc_AM_GetManageableAppLicenses (
@UserKey nvarchar(255) = null
,@SearchString nvarchar(1024) = null
,@PageSize int
,@StartingLicenseNumber int
,@RangeStart varbinary(529) OUTPUT
,@RangeEnd varbinary(529) OUTPUT
);
```

@UserKey: The key of the marketplace license director requesting the marketplace licenses. If a SharePoint Farm Administrator is doing the request, this value MUST be NULL. The value MUST be a **UserKey**(section 2.2.1.6).

@SearchString: The string that specifies a sub-string for the app titles of the marketplace licenses to be returned. The value MUST be a **SearchString** (section 2.2.1.24).

@PageSize: The maximum number of records to be returned. The value MUST be a **PageSize** (section 2.2.1.21).

@StartingLicenseNumber: The index of the first marketplace license to be returned among the whole list of qualifying marketplace licenses based on an implementation specific ordering. The value MUST be a **StartingRowNumber** (section 2.2.1.19).

@RangeStart: The beginning of the data range in which the results will be looked for. The value MUST be a **RangeStart** (section 2.2.1.22).

@RangeEnd: The end of the data range in which the results will be looked for. The value MUST be a **RangeEnd** (section 2.2.1.23).

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return a Manageable App Licenses.

This stored procedure MUST return an Item Count.

3.2.5.17 proc_AM_GetManageableAppLicensesForApp

The **proc_AM_GetManageableAppLicensesForApp** stored procedure is called to get a specified number of marketplace licenses for a given app. If the *@UserKey* is not NULL, the returned marketplace licenses MUST have a marketplace license director with the given key.

```
PROCEDURE proc_AM_GetManageableAppLicensesForApp (
@UserKey nvarchar(255) = null
,@CompositePartitionKey varbinary(529)
,@LimitCount int
,@ErrorCode int OUTPUT
);
```

@UserKey: The key of the marketplace license director requesting the marketplace licenses. If a SharePoint Farm Administrator is doing the request, this value MUST be NULL. The value MUST be a **UserKey**(section 2.2.1.6).

@CompositePartitionKey: The composite partition key identifying the app product identifier and the site subscription identifier of the app to get marketplace licenses for. The value MUST be an **AppLicensingCompositePartitionKey** (section 2.2.3.2).

@LimitCount: The maximum number of marketplace licenses to be returned. The value MUST be a **LimitCount** (section 2.2.1.20).

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

Value	Description
-1000	The *@CompositePartitionKey* value is outside the data range of the protocol server.
0	No errors encountered.

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return a Manageable App Licenses.

3.2.5.18 proc_AM_GetManageableAppLicensesForProductIds

The **proc_AM_GetManageableAppLicensesForProductIds** stored procedure is called to get the marketplace licenses for a list of composite partition keys.

Upon return from this stored procedure, the *@RangeStart* and *@RangeEnd* parameters MUST be set according to the following:

· If *@RangeStart* is greater than the end point of the protocol server data range, then *@RangeStart* and *@RangeEnd* MUST be set to the protocol server data range end point.

· If *@RangeEnd* is less than the start point of the protocol server data range, then *@RangeStart* and *@RangeEnd* MUST be set to the protocol server data range start point.

· If *@RangeStart* and *@RangeEnd* intersect with the protocol server data range, then *@RangeStart* MUST be set to the maximum of the *@RangeStart* value and protocol server data range start point, and *@RangeEnd* MUST be set to the minimum of the *@RangeEnd* value and the protocol server data range end point.

· NULL is the maximum value for any data range point comparison.

```
PROCEDURE proc_AM_GetManageableAppLicensesForProductIds (
@CompositePartitionKeys tvpArrayLicenseCompositeKeys
,@RangeStart varbinary(529) OUTPUT
,@RangeEnd varbinary(529) OUTPUT
);
```

@CompositePartitionKeys: The list of composite partition keys where each one of them is identifying an app product identifier, site subscription identifier pair for an app. The value MUST be a **Composite Partition Key List (App)** (section 2.2.7.2).

@RangeStart: The beginning of the data range in which the results will be looked for. The value MUST be a **RangeStart** (section 2.2.1.22).

@RangeEnd: The end of the data range in which the results will be looked for. The value MUST be a **RangeEnd** (section 2.2.1.23).

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return a Manageable App Licenses.

3.2.5.19 proc_AM_GetManageableApps

The **proc_AM_GetManageableApps** stored procedure is called to get all apps which have at least one marketplace license that a given marketplace license director can manage. Information about the marketplace license is also returned along with each app. If more than one marketplace license can be managed by the given marketplace license director for a particular app then they MUST be sorted and the marketplace license information returned MUST be for the first one in that ordered set.

The marketplace licenses MUST be sorted in ascending order by the license rank and then descending order by the acquisition date. The license rank is determined by the following (the current time is measured by the clock of the computer hosting the protocol server):

Rank	Marketplace license
1	Token expiry date is greater than or equal to the current time and marketplace license type is *PerpetualAllUsers*.
2	Token expiry date is greater than or equal to the current time and marketplace license type is *PerpetualMultiUser*.
3	Expiration date is greater than or equal to the current time, token expiry date is greater than or equal to the current time and marketplace license type is *TrialAllUsers*.
4	Expiration date is greater than or equal to the current time, token expiry date is greater than or equal to the current time and marketplace license type is *TrialMultiUser*.

Rank	Marketplace license
5	Token expiry date is less than the current time and marketplace license type is *PerpetualAllUsers*.
6	Token expiry date is less than the current time and marketplace license type is *PerpetualMultiUser*.
7	Expiration date is greater than or equal to the current time, token expiry date is less than the current time and marketplace license type is *TrialAllUsers*.
8	Expiration date is greater than or equal to the current time and token expiry date is less than the current time and marketplace license type is *TrialMultiUser*.
9	Expiration date is less than the current time, token expiry date is greater than or equal to the current time and marketplace license type is *TrialAllUsers*.
10	Expiration date is less than the current time, token expiry date is greater than or equal to the current time and marketplace license type is *TrialMultiUser*.
11	Expiration date is less than the current time, token expiry date is less than the current time and marketplace license type is *TrialAllUsers*.
12	Expiration date is less than the current time, token expiry date is less than the current time and marketplace license type is *TrialMultiUser*.

Upon return from this stored procedure, the **@RangeStart** and **@RangeEnd** parameters MUST be set according to the following:

· If *@RangeStart* is greater than the end point of the protocol server data range, then *@RangeStart* and *@RangeEnd* MUST be set to the protocol server data range end point.

· If *@RangeEnd* is less than the start point of the protocol server data range, then *@RangeStart* and *@RangeEnd* MUST be set to the protocol server data range start point.

· If *@RangeStart* and *@RangeEnd* intersect with the protocol server data range, then *@RangeStart* MUST be set to the maximum of the *@RangeStart* value and protocol server data range start point, and *@RangeEnd* MUST be set to the minimum of the *@RangeEnd* value and the protocol server data range end point.

· NULL is the maximum value for any data range point comparison.

```
PROCEDURE proc_AM_GetManageableApps (
@UserKey nvarchar(255)
,@MaxItemCountToFetch int
,@RangeStart varbinary(529) OUTPUT
,@RangeEnd varbinary(529) OUTPUT
);
```

@UserKey: The key of the marketplace license director. The value MUST be a **UserKey** (section 2.2.1.6). The value MUST NOT be NULL.

@MaxItemCountToFetch: The maximum number of apps to be returned. The value MUST be a **LimitCount** (section 2.2.1.20).

@RangeStart: The beginning of the data range in which the results will be looked for. The value MUST be a **RangeStart** (section 2.2.1.22).

@RangeEnd: The end of the data range in which the results will be looked for. The value MUST be a **RangeEnd** (section 2.2.1.23).

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return a Manageable Apps.

3.2.5.20 proc_AM_GetUserAssignmentsForLicenseById

The **proc_AM_GetUserAssignmentsForLicenseById** stored procedure is called to get all marketplace license user assignments for a specified marketplace license. If *@UserKey* is not NULL, the specified marketplace license MUST have a marketplace license director with that key for marketplace license user assignments to be returned. Otherwise an empty result set MUST be returned. If *@SearchedUsers* is not an empty list, user keys in the resulting marketplace license user assignments MUST also appear in the *@SearchedUsers* list.

```
PROCEDURE proc_AM_GetUserAssignmentsForLicenseById (
@LicenseId uniqueidentifier
,@CompositePartitionKey varbinary(33)
,@UserKey nvarchar(255)
,@PageSize int
,@StartingUserNumber int
,@SearchedUsers tvpUserList
,@ErrorCode int OUTPUT
);
```

@LicenseId: The identifier representing a marketplace license. The value MUST be a **LicenseId** (section 2.2.1.2).

@CompositePartitionKey: The composite partition key identifying the app product identifier and the site subscription identifier for the app of the marketplace license. The value MUST be an **AppLicensingCompositePartitionKey** (section 2.2.3.2).

@UserKey: The key of the marketplace license director requesting the marketplace license user assignments. If a SharePoint Farm Administrator is doing the request, this value MUST be NULL. The value MUST be a **UserKey** (section 2.2.1.6).

@PageSize: The maximum number of marketplace license user assignments to be returned. The value MUST be a **PageSize** (section 2.2.1.21).

@StartingUserNumber: The index of the first marketplace license user assignment to be returned among the whole list of qualifying marketplace license user assignments based on an implementation specific ordering. The value MUST be a **StartingRowNumber** (section 2.2.1.19).

@SearchedUsers: The set of users for which to filter the resulting list. This value MUST be a **User List** (section 2.2.7.1).

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

Value	Description
-1000	The *@CompositePartitionKey* value is outside the data range of the protocol server.

Value	Description
-8	The specified marketplace license does not exist.
-4	The specified user is not a marketplace license director for the specified marketplace license.
0	No errors encountered.

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return a License Users

This stored procedure MUST return an Item Count

3.2.5.21 proc_AM_ImportLicense

The **proc_AM_ImportLicense** stored procedure is called to save the marketplace license and app information by either:

- Creating a new marketplace license and/or app information record in the protocol server with these values or
- Updating the existing marketplace license and/or app information record with the site subscription identifier and the app product identifier matching with those encoded in *@CompositePartitionKey* and an app purchaser matching with the *@PurchaserIdentity.*

If and only if the following conditions are satisfied:

- *@MaxUserCount* is non-null and positive if *@LicenseType* is equal to 0 or 2.
- *@MaxUserCount* is null if *@LicenseType* is equal to 1 or 3.
- *@ExpirationDate* is non-null if *@LicenseType* is equal to 2 or 3.
- *@ExpirationDate* is null if *@LicenseType* is equal to 0 or 1.

Besides saving the given information, **proc_AM_ImportLicense** performs the following actions as well:

- If *@TokenExpiryDate* is sooner than 10 days, as measured by the clock of the computer hosting the protocol server, transaction approval status of the marketplace license MUST be marked as pending. In this case automatic renewal readiness status of the marketplace license MUST be set to not-ready at the time of the stored procedure call and MUST be set to ready 5 hours before *@TokenExpiryDate*. If *@TokenExpiryDate* is 10 days or more later, as measured by the clock of the computer hosting the protocol server, transaction approval status of the marketplace license is set to non-pending. In this case automatic renewal readiness status of the marketplace license MUST be set to not-ready at the time of the stored procedure call and MUST be set to ready 72 hours before *@TokenExpiryDate* in an implementation specific way. (see section 3.2.1 for transaction approval status and automatic renewal readiness status of marketplace licenses).
- If *@LicenseType* is equal to 0 or 2, the user represented by *@UserIdentity* and *@UserKey* MUST be assigned for the marketplace license (if not assigned already) as long as maximum user limit for the marketplace license is not violated.

- The user represented by *@UserIdentity* and *@UserKey* is assigned as a marketplace license director for the marketplace license (if not assigned already).
- If a marketplace license already existing on the protocol server is getting updated:
  - If *@LicenseType* is equal to 0 or 2, the license type of the already existing marketplace license is equal to 0 or 2 as well and *@MaxUserCount* is less than the number of already assigned users for the marketplace license; then excess users MUST be removed. The removed users are the ones who are assigned latest for the marketplace license. In this case the user represented by *@UserIdentity* and *@UserKey* MUST NOT be assigned for the marketplace license as a user.
  - If *@LicenseType* is equal to 1 or 3 and there exists user assignments for the updated marketplace license, all user assignments on the marketplace license MUST be removed.

```
PROCEDURE proc_AM_ImportLicense (
@CompositePartitionKey varbinary(33)
,@LicenseId uniqueidentifier
,@AppName nvarchar(1024)
,@UserIdentity nvarchar(255)
,@UserKey nvarchar(255)
,@PurchaserIdentity nvarchar(16)
,@LicenseType tinyint
,@OmexLicenseType tinyint
,@MaxUserCount int
,@ExpirationDate datetime
,@AssetId nvarchar(14)
,@DeploymentId uniqueidentifier
,@LicenseAcquisitionDate datetime
,@TokenExpiryDate datetime
,@ContentMarket nvarchar(10)
,@BillingMarket nvarchar(2)
,@IconUrl nvarchar(255)
,@ProviderName nvarchar(255)
,@RawXMLEntitlementToken nvarchar(512)
,@ErrorCode int OUTPUT
);
```

@CompositePartitionKey: The composite partition key identifying the app product identifier and the site subscription identifier of the app for which the marketplace license is issued. The value MUST be an **AppLicensingCompositePartitionKey** (section 2.2.3.2).

@LicenseId: The identifier of the marketplace license. If the marketplace license already exists in the protocol server, the given *@LicenseId* value MUST be discarded. This value MUST be a **LicenseId** (section 2.2.1.2).

@AppName: The title of the app. This value MUST be an **AppName** (section 2.2.1.4).

@UserIdentity: The name of the security principal (2) representing the user making the call. This value MUST be a **UserIdentity** (section 2.2.1.5).

@UserKey: The key of the security principal (2) representing the user making the call. This value MUST be a **UserKey** (section 2.2.1.6). This value MUST NOT be NULL.

@PurchaserIdentity: The identifier of the user who purchased this marketplace license. This value must be a **PurchaserIdentity** (section 2.2.1.8).

@LicenseType: The marketplace license type. The value MUST be a **LicenseType** (section 2.2.2.2).

@OmexLicenseType: The commercial type of the marketplace license. This value MUST be an **OmexLicenseType** (section 2.2.2.3).

@MaxUserCount: The maximum number of marketplace license user assignments allowed for this marketplace license. This value MUST be -1 if *@LicenseType* value is 1 or 3, and a positive integer if *@LicenseType* value is 2 or 4.

@ExpirationDate: The trial expiration date of this marketplace license. The value MUST be an **ExpirationDate** (section 2.2.1.11).

@AssetId: The marketplace asset identifier of the app for which this marketplace license is issued. This value MUST be an **AssetId** (section 2.2.1.3).

@DeploymentId: The marketplace deployment identifier to which this marketplace license is tied. This value MUST be a **DeploymentId** (section 2.2.1.17).

@LicenseAcquisitionDate: Date and time of the issuing of the marketplace license. This value MUST be a **LicenseAcquisitionDate** (section 2.2.1.9).

@TokenExpiryDate: Expiration date of the token of the marketplace license. This value MUST be a **TokenExpiryDate** (section 2.2.1.12).

@ContentMarket: The marketplace content market of the marketplace license. This value MUST be a **ContentMarket** (section 2.2.1.28).

@BillingMarket: The marketplace billing market of the marketplace license. This value MUST be a **BillingMarket** (section 2.2.1.29).

@IconUrl: The location of the icon image of the app. This value MUST be an **IconUrl** (section 2.2.1.30).

@ProviderName: The name of the publisher of the app. This value MUST be a **ProviderName** (section 2.2.1.31).

@RawXMLEntitlementToken: The marketplace license token. This value MUST be a **RawXMLEntitlementToken** (section 2.2.1.37).

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

Value	Description
-1000	The *@CompositePartitionKey* value is outside the data range of the protocol server.
-16	*@MaxUserCount* is non-positive and *@LicenseType* is equal to 0 or 2.
-10	*@ExpirationDate* is not null and *@LicenseType* is equal to 0 or 1.
-9	*@ExpirationDate* is null and *@LicenseType* is equal to 2 or 3.
-3	*@MaxUserCount* is not null and *@LicenseType* is equal to 1 or 3.
-2	*@MaxUserCount* is null and *@LicenseType* is equal to 0 or 2.
0	No errors encountered.

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return a Licenses.

3.2.5.22 proc_AM_PutAppPrincipal

The **proc_AM_PutAppPrincipal** stored procedure creates or updates an app principal. If an app principal with the specified composite partition key already exists and *@AuthoritySource* is 1, the existing app principal MUST be updated. If such an app principal does not exist, one MUST be created with the given parameter values.

```
PROCEDURE proc_AM_PutAppPrincipal (
@CompositePartitionKey varbinary(529)
,@Title nvarchar(255)
,@RedirectUrl nvarchar(255)
,@Realm nvarchar(255)
,@AuthoritySource bit
,@RequestGuid uniqueidentifier = null OUTPUT
);
```

@CompositePartitionKey: The composite partition key identifying the app principal identifier and the site subscription identifier of the app principal. The value MUST be an **AppRegistryCompositePartitionKey** (section 2.2.1.1).

@Title: The title of the app principal. This value MUST be an **AppPrincipalTitle**(section 2.2.1.39).

@RedirectUrl: The redirect URL of the app principal. This value MUST be a **RedirectUrl** (section 2.2.1.33).

@Realm: The app principal realm of the app principal. This value MUST be a **Realm** (section 2.2.1.34).

@AuthoritySource: Specifies whether the source of the information for the app can be used to update an existing app principal. This value MUST be an **IsUpdateAllowed** (section 2.2.1.35).

@RequestGuid: A GUID uniquely identifying the current request. The value MUST be a RequestGuid (section 2.2.1.1). <4>

Return Values: An integer which MUST be in the following table.

Value	Description
-1000	The *@CompositePartitionKey* value is outside the data range of the protocol server.
0	No errors encountered.

Result Sets:

This stored procedure MUST NOT return any result sets.

3.2.5.23 proc_AM_PutAppPrincipalPerm

The **proc_AM_PutAppPrincipalPerm** stored procedure is called to add or update an app principal permission for an app principal. If such an app principal does not exist, one MUST be created. If

such an app principal permission does not exist for the app principal, one MUST be created. When an app principal permission is updated, its version MUST be incremented and returned as @NewVersion.

```
PROCEDURE proc_AM_PutAppPrincipalPerm (
@CompositePartitionKey varbinary(529)
,@Title nvarchar(255)
,@RedirectUrl nvarchar(255)
,@Realm nvarchar(255)
,@AuthoritySource bit
,@ProviderTypeId uniqueidentifier
,@Perm varbinary(max)
,@Version int
,@NewVersion int OUTPUT
,@RequestGuid uniqueidentifier = null OUTPUT
);
```

@CompositePartitionKey: The composite partition key identifying the app principal identifier and the site subscription identifier of the app principal. This value MUST be an **AppRegistryCompositePartitionKey** (section 2.2.3.1).

@Title: The title of the app principal. This value MUST be an **AppPrincipalTitle** (section 2.2.1.39).

@RedirectUrl: The app principal identifier redirect URL for this app principal. This value MUST be a **RedirectUrl** (section 2.2.1.33).

@Realm: The app principal realm of the app principal. This value MUST be a **Realm** (section 2.2.1.34).

@AuthoritySource: Specifies whether the source of the information for the app can be used to update an existing app principal. This value MUST be an **IsUpdateAllowed** (section 2.2.1.35)

@ProviderTypeId: The unique identifier of an app principal permission provider. This value MUST be a **ProviderTypeId** (section 2.2.1.32).

@Perm: Implementation specific representation of the app principal permission. This value MUST be a **Permission** (section 2.2.1.40).

@Version: The current app principal permission version. This value MUST be a **Version** (section 2.2.1.41).

@NewVersion: The new app principal permission version. This value MUST be a **Version** (section 2.2.1.41).

@RequestGuid: A GUID uniquely identifying the current request. The value MUST be a RequestGuid (section 2.2.1.1). <5>

Return Values: An integer which MUST be in the following table.

Value	Description
-1000	The *@CompositePartitionKey* value is outside the data range of the protocol server.
-1	Version conflict.
0	No error encountered.

Result Sets:

This stored procedure MUST NOT return any result sets.

3.2.5.24 proc_AM_RegisterOAuthAppIdWithProduct

The **proc_AM_RegisterOAuthAppIdWithProduct** stored procedure is called to register the app principal which has the specified app principal identifier with the app which has specified app product identifier and the site subscription identifier, if such a registration does not already exist (for encoding details see section 2.2.3.2, for the concept of registration see section 3.2.1). The app instance identifier associated with the registration MUST be set to *@AppInstanceId*. If such a registration already exists, this stored procedure call MUST NOT change anything.

```
PROCEDURE proc_AM_RegisterOAuthAppIdWithProduct (
@CompositePartitionKey varbinary(33)
,@AppId nvarchar(256)
,@AppInstanceId uniqueidentifier
,@ErrorCode int OUTPUT
);
```

@CompositePartitionKey: The composite partition key identifying the app product identifier and the site subscription identifier of the app to be registered. The value MUST be an **AppLicensingCompositePartitionKey** (section 2.2.3.2).

@AppId: The app principal identifier of the app principal to be registered. The value MUST be an **AppId** (section 2.2.1.18).

@AppInstanceId: The app instance identifier to be associated with the registration. The value MUST be an **AppInstanceIdentifier** (section 2.2.1.36).

@ErrorCode: An integer which MUST be in the following table.

Value	Description
-1000	The *@CompositePartitionKey* value is outside the data range of the protocol server.
0	No errors encountered.

Return Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.25 proc_AM_RemoveAppPrincipalPerms

The **proc_AM_RemoveAppPrincipalPerms** stored procedure is called to remove a set of app principal permissions from an app principal.

```
PROCEDURE proc_AM_RemoveAppPrincipalPerms (
@CompositePartitionKey varbinary(529)
,@ProviderTypeId uniqueidentifier
,@RequestGuid uniqueidentifier = null OUTPUT
);
```

@CompositePartitionKey: The composite partition key identifying the app principal identifier and the site subscription identifier of the app principal. This value MUST be an **AppRegistryCompositePartitionKey** (section 2.2.3.1).

@ProviderTypeId: The identifier for the app principal permission provider of the app principal. If this value is NULL, the app principal permissions for all permission providers MUST be removed.

@RequestGuid: A GUID uniquely identifying the current request. The value MUST be a RequestGuid (section 2.2.1.1). <6>

Return Values: An integer which MUST be in the following table.

Value	Description
-1000	The *@CompositePartitionKey* value is outside the data range of the protocol server.
0	No errors encountered.

Result Sets:

This stored procedure MUST NOT return any result sets.

3.2.5.26 proc_AM_RemoveLicenseDirectors

The **proc_AM_RemoveLicenseDirectors** stored procedure is called to remove a set of marketplace license directors from the specified marketplace license. The stored procedure returns the set of specified users for whom there is no marketplace license director given the specific marketplace license.

```
PROCEDURE proc_AM_RemoveLicenseDirectors (
@Users tvpUserList
,@LicenseId uniqueidentifier
,@CompositePartitionKey varbinary(33)
,@UserKey nvarchar(255)
,@ErrorCode int OUTPUT
);
```

@Users: The set of users to remove marketplace license directors for. The value MUST be a **User List** (section 2.2.7.1).

@LicenseId: The identifier of the marketplace license. The value MUST be a **LicenseId** (section 2.2.1.2).

@CompositePartitionKey: The composite partition key identifying the app product identifier and the site subscription identifier for the app of the marketplace license. The value MUST be an **AppLicensingCompositePartitionKey** (section 2.2.3.2).

@UserKey: The key of the marketplace license director removing the marketplace licenses directors. If a SharePoint Farm Administrator is doing the request, this value MUST be NULL. The value MUST be a **UserKey**(section 2.2.1.6).

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

Value	Description
-1000	The *@CompositePartitionKey* value is outside the data range of the protocol server.
-14	The marketplace license type is neither *PerpetualMultiUser* nor *TrialMultiUser*.
-8	The specified marketplace license does not exist.
-4	The specified user is not a marketplace license director of the specified marketplace license.
0	No errors encountered.

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return a License Directors.

3.2.5.27 proc_AM_RemoveRegisteredOAuthAppId

The **proc_AM_RemoveRegisteredOAuthAppId** stored procedure is called to remove the registration for the app principal with the specified app principal identifier and the app with the specified app product identifier, site subscription identifier pair. The registration to be removed MUST be associated with the specified app instance identifier.

```
PROCEDURE proc_AM_RemoveRegisteredOAuthAppId (
@CompositePartitionKey varbinary(33)
,@AppId nvarchar(256)
,@AppInstanceId uniqueidentifier
,@ErrorCode int OUTPUT
);
```

@CompositePartitionKey: The composite partition key identifying the app product identifier and the site subscription identifier for the app of the registration to be removed. The value MUST be an **AppLicensingCompositePartitionKey** (section 2.2.3.2).

@AppId: The app principal identifier for the app principal of the registration to be removed. The value MUST be an **AppId** (section 2.2.1.18).

@AppInstanceId: The app instance identifier associated with the registration to be removed. The value MUST be an **AppInstanceIdentifier** (section 2.2.1.36).

@ErrorCode: An integer which MUST be in the following table.

Value	Description
-1000	The *@CompositePartitionKey* value is outside the data range of the protocol server.
0	No errors encountered.

Return Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.28 proc_AM_RemoveUserLicenseAssignments

The **proc_AM_RemoveUserLicenseAssignments** stored procedure is called to remove a set of marketplace license user assignments from a marketplace license on behalf of a marketplace license director or a SharePoint Farm Administrator. The stored procedure returns the set of specified users for whom there is no marketplace license user assignment given the specific marketplace license.

```
PROCEDURE proc_AM_RemoveUserLicenseAssignments (
@Users tvpUserList
,@LicenseId uniqueidentifier
,@CompositePartitionKey varbinary(33)
,@UserKey nvarchar(255)
,@ErrorCode int OUTPUT
);
```

@Users: The set of users to remove marketplace license user assignments for. This value MUST be a **User List** (section 2.2.7.1).

@LicenseId: The identifier of the marketplace license. This value MUST be a **LicenseId**(section 2.2.1.2).

@CompositePartitionKey: The composite partition key identifying the app product identifier and the site subscription identifier for the app of the marketplace license. The value MUST be an **AppLicensingCompositePartitionKey** (section 2.2.3.2).

@UserKey: The key of the marketplace license director removing the marketplace licenses user assignments. If a SharePoint Farm Administrator is doing the request, this value MUST be NULL. The value MUST be a **UserKey**(section 2.2.1.6).

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

Value	Description
-1000	The @*CompositePartitionKey* value is outside the data range of the protocol server.
-14	The marketplace license type is neither *PerpetualMultiUser* nor *TrialMultiUser*.
-8	The specified marketplace license does not exist.
-7	The protocol server encountered an implementation-specific data integrity violation.
-4	The specified user is not a marketplace license director of the specified marketplace license.
0	No errors encountered.

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return a License Users.

3.2.5.29 proc_AM_SetDeploymentId

The **proc_AM_SetDeploymentId** stored procedure is called to set the marketplace deployment identifier associated with the specified site subscription identifier.

```
PROCEDURE proc_AM_SetDeploymentId (
@CompositePartitionKey binary(17)
,@NewDeploymentId uniqueidentifier
,@ErrorCode int OUTPUT
);
```

@CompositePartitionKey: The composite partition key identifying the site subscription identifier for which to set the marketplace deployment identifier. The value MUST be a **DeploymentIdCompositePartitionKey** (section 2.2.3.3).

@NewDeploymentId: The new marketplace deployment identifier to be associated with the specified site subscription identifier. The value MUST be a **DeploymentId** (section 2.2.1.17).

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

Value	Description
-1000	The *@CompositePartitionKey* value is outside the data range of the protocol server.
0	No errors encountered.

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return a Deployment ID.

3.2.5.30 proc_AM_UpdateAppInstanceDeploymentData

The **proc_AM_UpdateAppInstanceDeploymentData** stored procedure is called to set the app instance identifier of a specified app principal, and the deployment data of that app instance.

```
PROCEDURE proc_AM_UpdateAppInstanceDeploymentData (
@CompositePartitionKey varbinary(529)
,@AppInstanceId uniqueidentifier
,@DeploymentData nvarchar(max)
,@RequestGuid uniqueidentifier = null OUTPUT
);
```

@CompositePartitionKey: The composite partition key identifying the app principal identifier and the site subscription identifier. This value MUST be an **AppRegistryCompositePartitionKey** (section 2.2.3.1).

@AppInstanceId: The app instance identifier. The value MUST be an **AppInstanceIdentifier** (section 2.2.1.36).

@DeploymentData: The deployment data. The value MUST be a **DeploymentData** (section 2.2.1.27).

@RequestGuid: A GUID uniquely identifying the current request. The value MUST be a RequestGuid (section 2.2.1.1). <7>

Return Values: An integer which MUST be in the following table.

Value	Description
-1000	The *@CompositePartitionKey* value is outside the data range of the protocol server.
0	No errors encountered.

Result Sets:

This stored procedure MUST NOT return any result sets.

3.2.5.31 proc_AM_UpdateAppPrincipalFlags

The **proc_AM_UpdateAppPrincipalFlags** stored procedure is called to change state flags of a specified app principal.

```
PROCEDURE proc_AM_UpdateAppPrincipalFlags (
@CompositePartitionKey varbinary(529)
,@Flag int
,@SetOrClearFlags bit
,@RequestGuid uniqueidentifier = null OUTPUT
);
```

@CompositePartitionKey: The composite partition key identifying the app principal identifier and the site subscription identifier of the app principal. This value MUST be an **AppRegistryCompositePartitionKey** (section 2.2.3.1).

@Flag: State flags to be modified. This value MUST be an **AppPrincipalFlags**(section 2.2.2.1).

@SetOrClearFlags: Whether to add or remove the flags in *@Flag*. If this value is set to 1, the state flags of the app principal MUST be set to the bitwise-or of its current flags and the flags in *@Flag*. Otherwise, the state flags of the app principal MUST be set to the bitwise-and of its current flags and the flags in *@Flag*.

@RequestGuid: A GUID uniquely identifying the current request. The value MUST be a RequestGuid (section 2.2.1.1). <8>

Return Values: An integer which MUST be in the following table.

Value	Description
-1000	The *@CompositePartitionKey* value is outside the data range of the protocol server.
0	No errors encountered.

Result Sets:

This stored procedure MUST NOT return any result sets.

3.2.5.32 proc_AM_GetManageableSubsetForProductIds

The **proc_AM_GetManageableSubsetForProductIds** stored procedure is called to get all the marketplace licenses which belong to an app with a composite partition key in the specified list of composite partition keys. If *@UserKey* is not NULL. All marketplace licenses returned MUST have the specified marketplace license director.

Upon return from this stored procedure, the *@RangeStart* and *@RangeEnd* parameters MUST be set according to the following:

· If *@RangeStart* is greater than the end point of the protocol server data range, then *@RangeStart* and *@RangeEnd* MUST be set to the protocol server data range end point.

· If *@RangeEnd* is less than the start point of the protocol server data range, then *@RangeStart* and *@RangeEnd* MUST be set to the protocol server data range start point.

· If *@RangeStart* and *@RangeEnd* intersect with the protocol server data range, then *@RangeStart* MUST be set to the maximum of the *@RangeStart* value and protocol server data range start point, and *@RangeEnd* MUST be set to the minimum of the *@RangeEnd* value and the protocol server data range end point.

· NULL is the maximum value for any data range point comparison.

```
PROCEDURE proc_AM_GetManageableSubsetForProductIds (
@UserKey nvarchar(255)
,@CompositePartitionKeyList tvpCompositePartitionKeyList
,@RangeStart varbinary(529) OUTPUT
,@RangeEnd varbinary(529) OUTPUT
);
```

@UserKey: The key of the marketplace license director requesting the marketplace licenses. If a SharePoint Farm Administrator is doing the request, this value MUST be NULL. The value MUST be a **UserKey**(section 2.2.1.6).

@CompositePartitionKeyList: The list of composite partition keys identifying the app product identifiers and site subscription identifiers of the apps. The value MUST be a **Composite Partition Key List (App)** (section 2.2.7.2).

@RangeStart: The beginning of the data range in which the results will be looked for. The value MUST be a **RangeStart** (section 2.2.1.22).

@RangeEnd: The end of the data range in which the results will be looked for. The value MUST be a **RangeEnd** (section 2.2.1.23).

Return Values: An integer which MUST be in the following table.

Value	Description
-1000	The *@CompositePartitionKey* value is outside the data range of the protocol server.
0	No errors encountered.

Result Sets:

This stored procedure MUST return a Manageable App Identifiers

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

3.3 Client Details

The protocol client acts as a client when it calls the back-end database server requesting processing of stored procedures and optionally caching some of the data retrieved by the stored procedures.

3.3.1 Abstract Data Model

None.

3.3.2 Timers

The protocol client MUST use timers to ensure that marketplace licenses are renewed regularly so that they do not expire or stay expired for long periods of time. For this purpose, the following timer is used:

- **Automatic license renewal timer:** It is used to trigger the license renewal timer job execution described in section 3.3.6.1, which is used to renew all marketplace licenses stored in the protocol server. This timer SHOULD be set to 1 hour and MUST NOT be set to a value less than or equal to 2 minutes. The timer MUST be reset when the job execution ends.

3.3.3 Initialization

A connection that uses the underlying protocol layers that are specified in section 1.4 MUST be established before using this protocol, as specified in [MS-TDS].

3.3.4 Higher-Layer Triggered Events

None.

3.3.5 Message Processing Events and Sequencing Rules

The protocol client handles each stored procedure with the same processing method of calling the stored procedure and waiting for the return code and any result sets that will be returned.

3.3.6 Timer Events

3.3.6.1 Automatic License Renewal Timer Timeout

The **Automatic License Renewal Timer Timeout** event executes the license renewal timer job, which MUST do the following in the order presented here:

1. Calling **proc_AM_AcquireEntitlementsForRenewal** (section 3.2.5.1) one or more times in such a way that all the marketplace licenses stored in the protocol server with **Ready** automatic renewal readiness status (section 3.2.1) at the time of the job execution are returned.

2. Applying an implementation specific renewal process on the marketplace licenses gathered in step 1 and categorizing them into the following four categories:

   - **Renewed**

   - **To be deleted**

- **To be retried**
- **To be preserved**

3. Calling **proc_AM_BulkImportLicense** (section 3.2.5.6) one or more times in such a way that all the marketplace licenses collected in the four categories after executing step 2 are put into the parameter lists of **proc_AM_BulkImportLicense** as follows:
   - Those in **Renewed** category take place in *@RenewedEntitlements*.
   - Those in **To be deleted** category take place in *@EntitlementsToBeDeleted*.
   - Those in **To be retried** category take place in *@ EntitlementsToBeRetried*.
   - Those in **To be preserved** category take place in *@ EntitlementsToBePreserved*.

3.3.7 Other Local Events

None.

4 Protocol Examples

This section provides specific example scenarios for manipulating marketplace licenses. These examples describe in detail the process of communication between the protocol client and the protocol server.

4.1 Import License

This scenario is initiated when the user wants to import a marketplace license for an app.

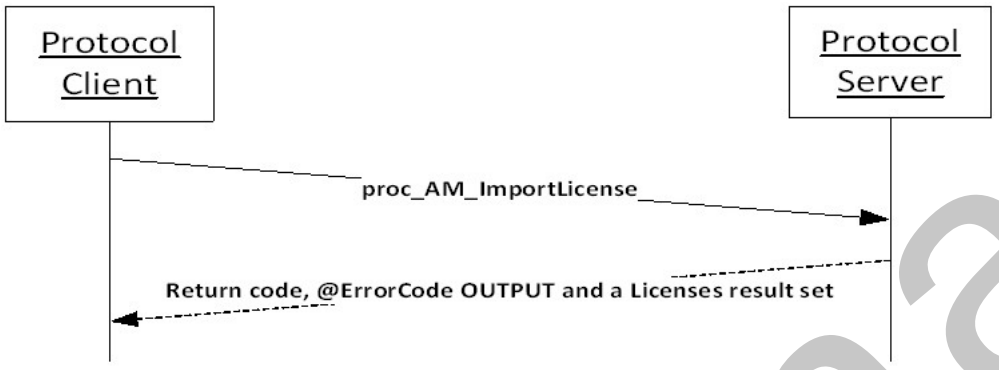

Figure 2: Import License

The following actions happen:

1. The protocol client sends a request to import the acquired marketplace license for an app by calling the **proc_AM_ImportLicense** (section 3.2.5.21) stored procedure.
2. The protocol server imports the new marketplace license.
3. The protocol server always returns 0 for the return code. It will return 0 for *@ErrorCode* output parameter if no errors were encountered and a **Licenses** (section 2.2.4.14) result set. If an error was encountered *@ErrorCode* output parameter will be set to reflect the type of error.

4.2 Get Manageable App Licenses

This scenario is initiated when the user wants to get all the marketplace licenses that he or she can manage.

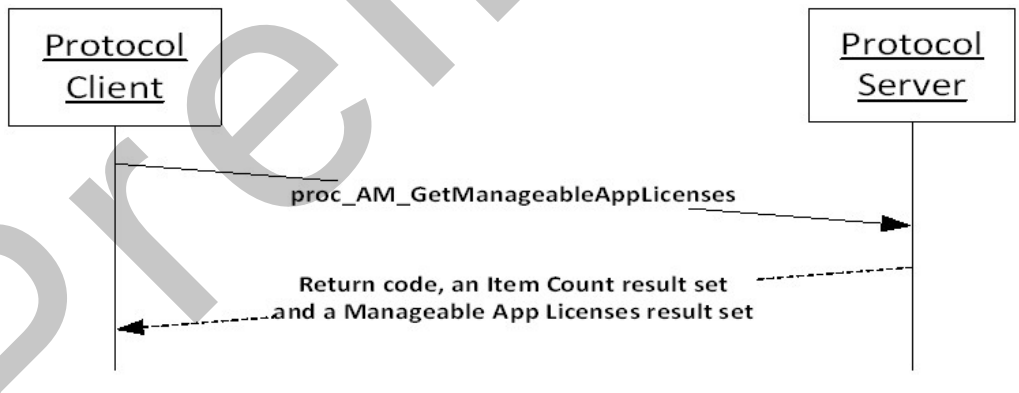

Figure 3: Get Manageable App Licenses

The following actions happen:

1. The protocol client sends a request to get the marketplace licenses that the user can manage by calling the **proc_AM_GetManageableAppLicenses** (section 3.2.5.16) stored procedure.

2. The protocol server always returns 0 for the return code. It will return a set of marketplace licenses the given user can manage. The number of records will be returned in an **Item Count** (section 2.2.4.11) result set and each record will be returned a **Manageable App Licenses** (section 2.2.4.15) result set.

4.3 Get Manageable App License with Identifier

This scenario is initiated when the user wants to get a specific marketplace license.

Figure 4: Get Manageable App Licenses with Identifier

The following actions happen:

1. The protocol client sends a request to get a particular marketplace license that the user can manage by calling the **proc_AM_GetManageableAppLicenseById** (section 3.2.5.15) stored procedure.

2. The protocol server always returns 0 for the return code. It will return 0 for *@ErrorCode* output parameter if no errors were encountered and a **Manageable App Licenses** (section 2.2.4.15) result set which contains the requested marketplace license. If an error was encountered *@ErrorCode* output parameter will be set to reflect the type of error.

4.4 Add User License Assignments

This scenario is initiated when the user wants to add a set of marketplace license user assignments to a marketplace license.

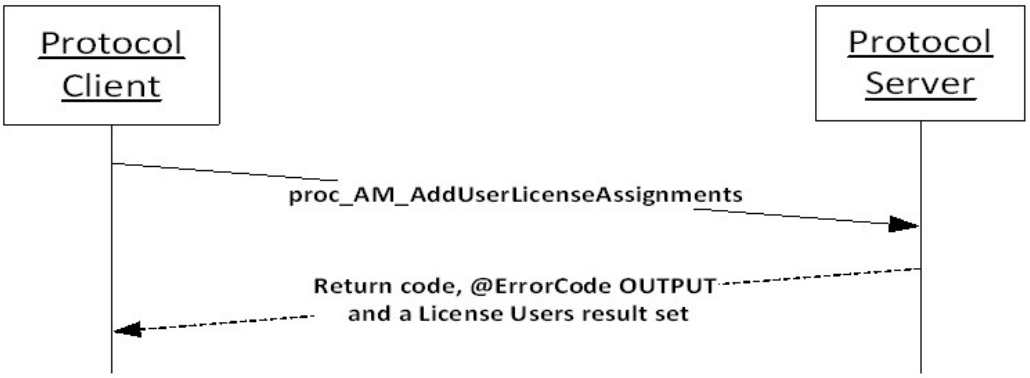

Figure 5: Add User License Assignments

The following actions happen:

1. The protocol client sends a request to assign a marketplace license to a set of users by calling the **proc_AM_AddUserLicenseAssignments** (section 3.2.5.4) stored procedure.

2. The protocol server always returns 0 for the return code. It returns the set of specified users who already had the specified marketplace license assigned to them in a **License Users** (section 2.2.4.13) result set. It will return 0 for *@ErrorCode* output parameter if no errors were encountered. If an error was encountered *@ErrorCode* output parameter will be set to reflect the type of error.

4.5 Get App Principal

This scenario is initiated when the user wants to get the set of app principals for a particular app.

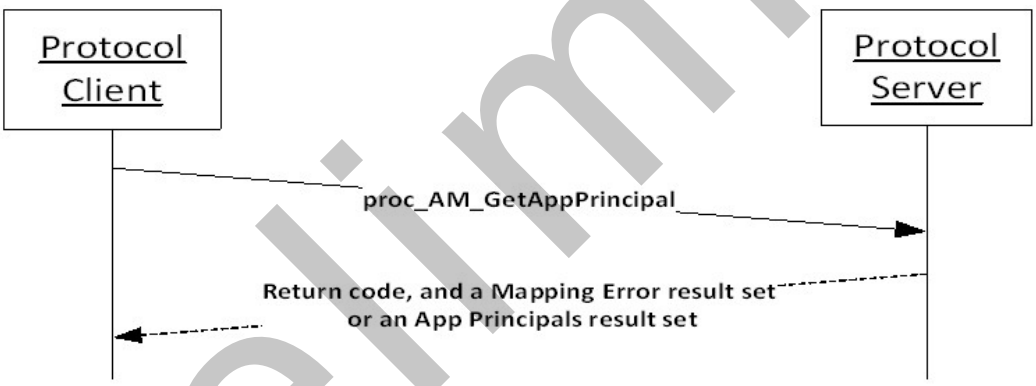

Figure 6: Get App Principal

The following actions happen:

1. The protocol client sends a request to get the app principal for an app by calling the **proc_AM_GetAppPrincipal** (section 3.2.5.9) stored procedure.

2. The protocol server always returns a 0 for the return code. If the composite partition key specified by the protocol client is outside the data range of the protocol server it will return a **Mapping Error** (section 2.2.4.2) result set. Otherwise, it will return an **App Principals** (section 2.2.4.5) result set.

4.6 Get Marketplace Deployment Identifier

This scenario is initiated when the user wants to get the marketplace deployment identifier associated with a particular site subscription.

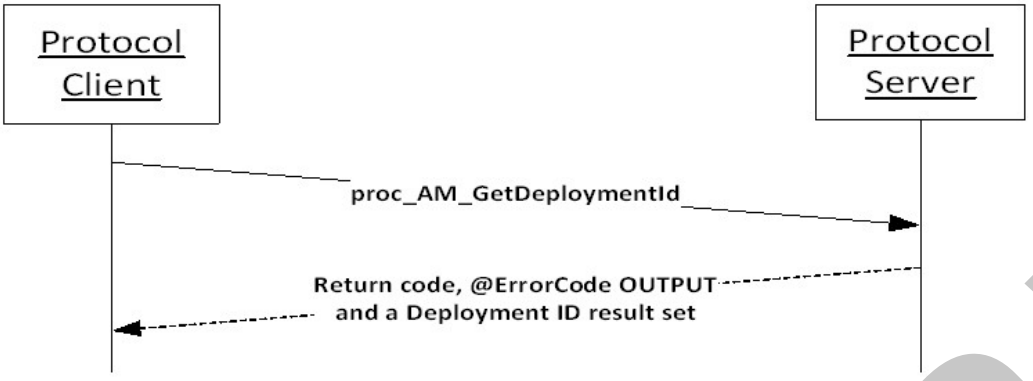

Figure 7: Get Marketplace Deployment Identifier

The following actions happen:

1. The protocol client sends a request to get the marketplace deployment identifier by calling the **proc_AM_GetDeploymentId** (section 3.2.5.12) stored procedure.

2. The protocol server always returns a 0 for the return code. It will return 0 for *@ErrorCode* output parameter if no errors were encountered and a **Deployment ID** (section 2.2.4.1) result set. If an error was encountered *@ErrorCode* output parameter will be set to reflect the type of error.

5 Security

5.1 Security Considerations for Implementers

Interactions with SQL are susceptible to tampering and other forms of security risks. Implementers are advised to sanitize input parameters for stored procedures before invoking the stored procedure.

There are no additional security considerations. Security assumptions for this protocol are documented in section 1.5.

5.2 Index of Security Parameters

None.

6 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

- Microsoft® SharePoint® Foundation 2013 Preview
- Microsoft® SharePoint® Server 2013 Preview

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.

<1> Section 3.2.5.8: In SharePoint Foundation 2013 Preview, this is always set to NULL.

<2> Section 3.2.5.9: In SharePoint Foundation 2013 Preview, this is always set to NULL.

<3> Section 3.2.5.10: In SharePoint Foundation 2013 Preview, this is always set to NULL.

<4> Section 3.2.5.22: In SharePoint Foundation 2013 Preview, this is always set to NULL.

<5> Section 3.2.5.23: In SharePoint Foundation 2013 Preview, this is always set to NULL.

<6> Section 3.2.5.25: In SharePoint Foundation 2013 Preview, this is always set to NULL.

<7> Section 3.2.5.30: In SharePoint Foundation 2013 Preview, this is always set to NULL.

<8> Section 3.2.5.31: In SharePoint Foundation 2013 Preview, this is always set to NULL.

7 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.

8 Index

E

F

G

H

I

L

M

N

O

P

R

S

T

U

V

X