(T14)討論 Index
CourseGUID: e48417fc-9db5-4e99-822c-706c5ccef6cc
==========
(T14)討論 Index
==========

==========

# 0. Summary

1.
Tables and Views can create Indexes to improve the performance of the query.
Indexes concept is similar to book index or table of content.
Firtly, look at index, then find out the data address,
and go to that address directly and find the data.
Without Indexes, you have to do Table Scan which means
search from first data row to last data rows.
Table Scan is no good for  the performance of the query.
2.
Types of Indexes in SQL server
2.1. Clustered
2.2. Nonclustered
2.3. Unique
2.4. Filtered
2.5. XML
2.6. Full Text
2.7. Spatial
2.8. Columnstore
2.9. Index with included columns
2.10. Index on computed columns
Here, we only discuss Clustered, Nonclustered, Unique Indexes.
3.
--DROP INDEX Gamer.PK__Gamer__3214EC0732501013;
This will return Erro
You can not use Query to drop Clustered Index
But you can drop Clustered Index in SSMS.
In SSMS, delete the index
Database Name --> Table Name --> Indexes
--> Right Click --> Delete
4.
Clustered Index V.S. NonClustered Index
4.1.
Clustered Index
4.1.1.
One table can only have ONE clustered index.

By default, SQL server will set Primary Key as the clustered index
if there is no clustered index yet at that time.
4.1.2.
A Clustered index is stored with table and
does not need additional disk space.
it determines the storage order of data physically in the table.
---
4.2.
NonClustered Index
4.2.1.
One table can have many NonClustered Index.
4.2.2.
A Non-Clustered index is in one place and
refer to another place which stores data physically.
Because it need to refer back to the table,
Clustered index is slightly faster than a non clustered index.
---
4.3.
A composite index is an index on two or more columns.
If you select ColmnA and ColumnB and
Both ColmnA and ColumnB are in the composite IndexA.
Then this is a covering query by the IndexA.
In this case, the data can simply be returned from the composite IndexA.
A Clustered Index always covers a query,
because it contains all data in a table.
5.
5.1.
Good at Index:
If we have Index in ColumnA,
Index is good for
WHERE, WHERE with DELETE/UPDATE, ORDER BY, GROUP BY in ColumnA
5.2.
Bad at Index:
5.2.1.
NonClustered Index need additional disk space.
5.2.2.
When there are a lof of data in the table,
then DELETE or UPDATE performace might be bad.
Because it need extra time to update Indexes.
6.
UNIQUE is a property which can be assigned to
both CLUSTERED and NON-CLUSTERED indexes.
UNIQUE property ensure there are no duplicate data.
E.g.
If the index has UNIQUE property which
contains 2 columns, LastName and FirstName.
That means UNIQUE property ensures
there are no two enties has the same LastName and FirstName.
By default of SQL server,
6.1.
When you create a PRIMARY KEY constraint
which automatically creates a unique clustered index.
6.2.
Add new UNIQUE CONSTRAINT will automatically add  UNIQUE NONCLUSTERED INDEX
Drop the  UNIQUE CONSTRAINT will automatically drop  UNIQUE NONCLUSTERED INDEX.
6.3.
if there are duplicate values in the Email column,
then you will have to do something
and ensure there is no duplicate values
before you set a UNIQUE constraint to Email Column.
7.
Discuss --WITH IGNORE_DUP_KEY;
--CREATE UNIQUE INDEX IX_Gamer_DepartmentID
--ON Gamer(DepartmentID)
--WITH IGNORE_DUP_KEY;

when ColumnA have a unique index or constraint,
then ColumnA ensures there is no duplicate data.
E.g.
If I try to insert 5 data rows,
but there are 2 data rows contain duplicates.
Then all 5 data rows will be rejected.
--WITH IGNORE_DUP_KEY;
In this case, it allow to ignore thoese 2 duplicate rows.
and only insert the rest 3 data rows.
8.
8.1.
Create Nonclustered Index Syntax1:
--CREATE INDEX IX_TableName_ColumnName
--ON TableName (ColumnName);
E.g.
--CREATE INDEX IX_Gamer_GameScore
--ON Gamer (GameScore ASC);
Create Nonclustered Index on GameScore
8.2.
Create Nonclustered Index Syntax2:
--CREATE NONCLUSTERED INDEX IX_TableName_ColumnName
--ON TableName (ColumnName);
E.g.
--CREATE NONCLUSTERED INDEX IX_Gamer_Email
--ON Gamer(Email);
Create Nonclustered Index on Email
9.
--sys.sp_helpconstraint V.S. sys.sp_helpindex
9.1.
--EXECUTE sys.sp_helpconstraint @objname = N'TableName';
Get the constraint information of the Table
9.2.
--EXECUTE sys.sp_helpindex @objname = N'TableName';
Or
--EXEC sp_helpindex N'TableName'
Get the Index information of the Table
E.g.
EXEC sp_helpindex N'Gamer';

==========

# 1. SSMS - Create/Delete Index

Create a new Index

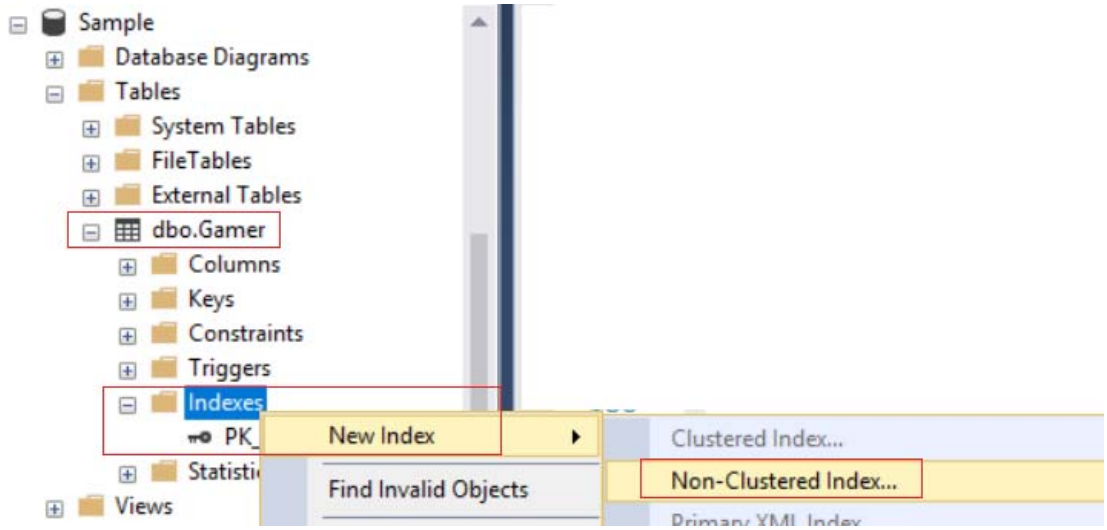
Database Name --> Table Name --> Indexes
New Index --> **Non-Clustered Index**

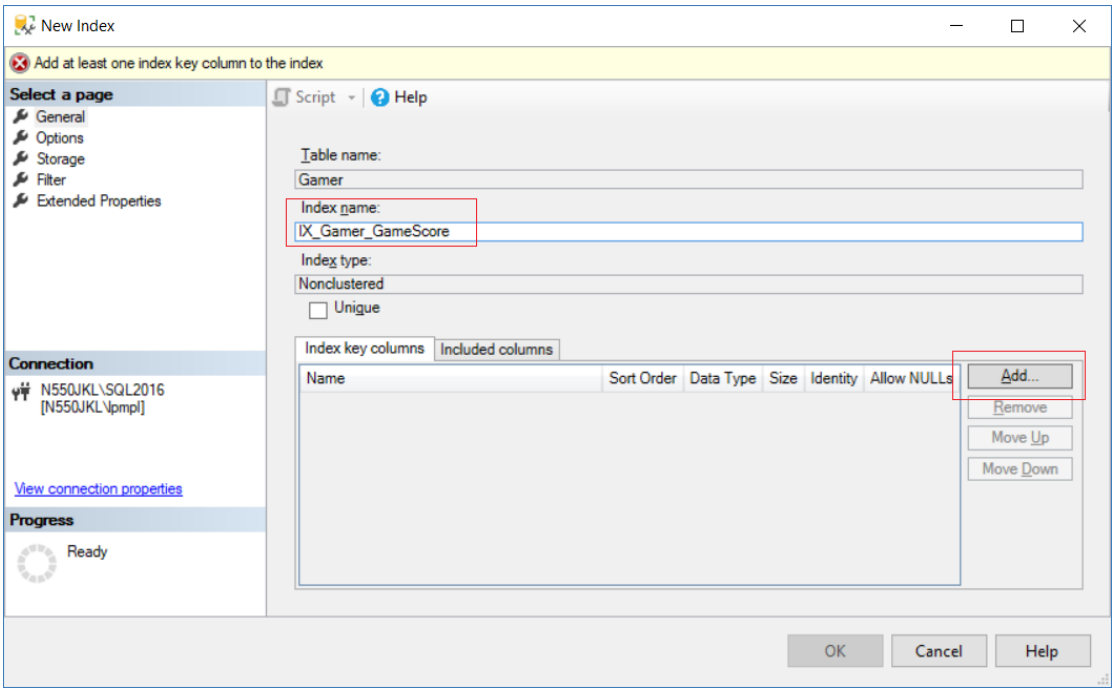
Index Name
**IX_Employee_AnnualSalary**
-->
Add
--> Select the Column Name
--> OK
--> OK

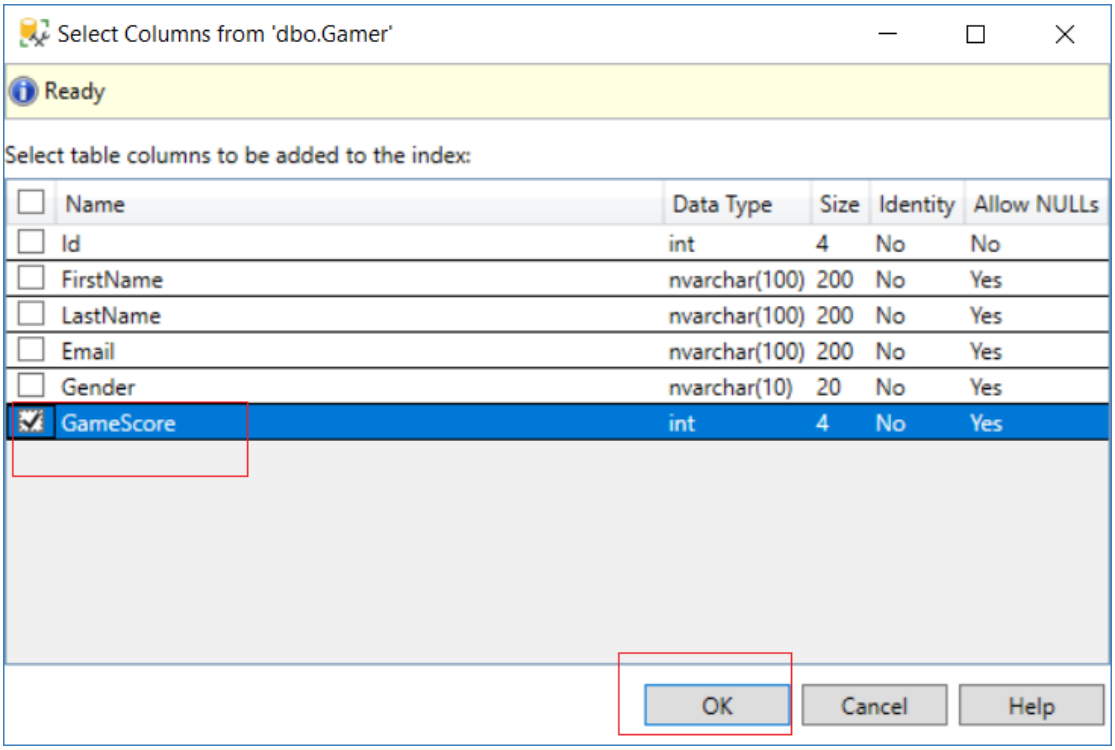
---

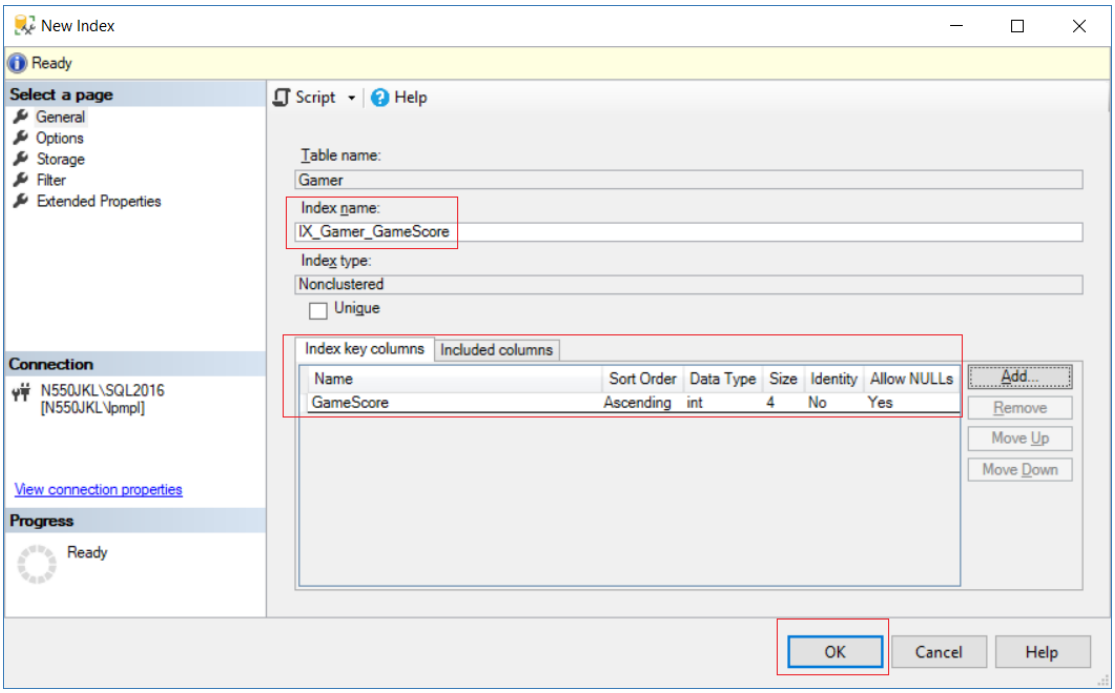
Delete/Disable an Index

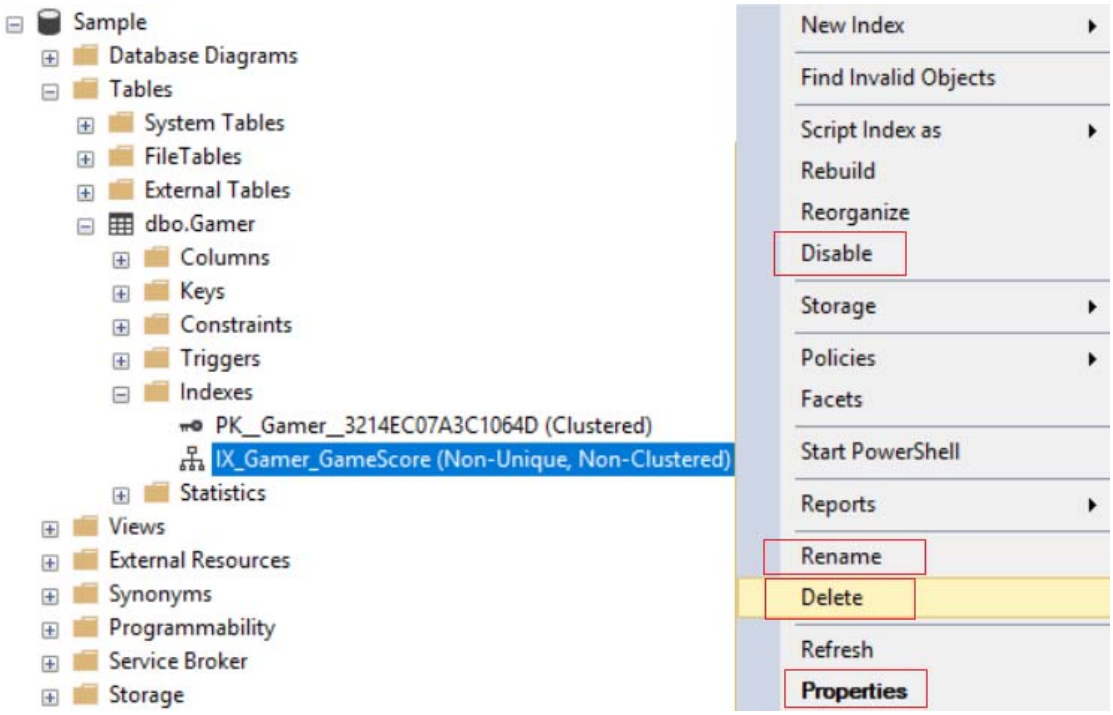
Database Name --> Table Name --> Indexes
--> Right Click --> Delete or Disable




Select Columns from 'dbo.Gamer'
Ready
Select table columns to be added to the index:
Name
Data Type
Size
Identity
Allow NULLs
Id
int
4
No
No
FirstName
nvarchar(100)
200
No
Yes
LastName
nvarchar(100)
200
No
Yes
Email
nvarchar(100)
200
No
Yes
Gender
nvarchar(10)
20
No
Yes
GameScore
int
4
No
Yes
OK
Cancel
Help


New Index
Ready
Select a page
General
Options
Storage
Filter
Extended Properties
Connection
N550JKL\SQL2016
[N550JKL\lpmpl]
View connection properties
Progress
Ready
Script
Help
Table name:
Gamer
Index name:
IX_Gamer_GameScore
Index type:
Nonclustered
Unique
Index key columns
Included columns
Name
Sort Order
Data Type
Size
Identity
Allow NULLs
GameScore
Ascending
int
4
No
Yes
Add...
Remove
Move Up
Move Down
OK
Cancel
Help




# 2. Query - Index

## 2.1. Create Sample Data

```
--=====================================================================
--T014_01_Create Sample Data
--=====================================================================
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE      TABLE_NAME = 'Gamer' ) )
    BEGIN
        TRUNCATE TABLE Gamer;
        DROP TABLE Gamer;
    END;
GO -- Run the previous command and begins new batch
CREATE TABLE Gamer
(
  Id INT PRIMARY KEY ,
  FirstName NVARCHAR(100) ,
  LastName NVARCHAR(100) ,
  Email NVARCHAR(100) ,
  Gender NVARCHAR(10) ,
  GameScore INT
);
GO -- Run the previous command and begins new batch
```

```sql
INSERT INTO Gamer
VALUES ( 4, 'First4', 'Last4', '4@4.com', 'Male', 43000 );
INSERT INTO Gamer
VALUES ( 2, 'First2', 'Last2', '2@2.com', 'Female', 44000 );
INSERT INTO Gamer
VALUES ( 1, 'First1', 'Last1', '1@1.com', 'Male', 43000 );
INSERT INTO Gamer
VALUES ( 5, 'First5', 'Last5', '5@5.com', 'Male', 42000 );
INSERT INTO Gamer
VALUES ( 3, 'First3', 'Last3', '3@3.com', 'Female', 41000 );
GO -- Run the previous command and begins new batch
SELECT *
FROM   Gamer;
GO -- Run the previous command and begins new batch
```

| | Id | FirstName | LastName | Email | Gender | GameScore |
|---|---|---|---|---|---|---|
| 1 | 1 | First1 | Last1 | 1@1.com | Male | 43000 |
| 2 | 2 | First2 | Last2 | 2@2.com | Female | 44000 |
| 3 | 3 | First3 | Last3 | 3@3.com | Female | 41000 |
| 4 | 4 | First4 | Last4 | 4@4.com | Male | 43000 |
| 5 | 5 | First5 | Last5 | 5@5.com | Male | 42000 |

## 2.2. Clustered Index

```sql
--=========================================================================
--T014_02_Clustered Index
--=========================================================================
/*
1.
Tables and Views can create Indexes to improve the performance of the query.
Indexes concept is similar to book index or table of content.
Firtly, look at index, then find out the data address,
and go to that address directly and find the data.
Without Indexes, you have to do Table Scan which means
search from first data row to last data rows.
Table Scan is no good for  the performance of the query.
2.
Types of Indexes in SQL server
2.1. Clustered
2.2. Nonclustered
2.3. Unique
2.4. Filtered
2.5. XML
2.6. Full Text
2.7. Spatial
2.8. Columnstore
2.9. Index with included columns
2.10. Index on computed columns
Here, we only discuss Clustered, Nonclustered, Unique Indexes.
3.
In SSMS, delete the index
Database Name --> Table Name --> Indexes
--> Right Click --> Delete
You are not allowed to delete index by query,
--DROP INDEX Gamer.PK__Gamer__3214EC0732501013;
but you may delete the index in SSMS.
4.
```

```
Clustered Index V.S. NonClustered Index
4.1.
Clustered Index
4.1.1.
One table can only have ONE clustered index.
By default, SQL server will set primary key Column as the clustered index
if there is no clustered index yet at that time.
4.1.2.
A Clustered index is stored with table and
does not need additional disk space.
it determines the storage order of data physically in the table.
-------
4.2.
NonClustered Index
4.2.1.
One table can have many NonClustered Index.
4.2.2.
A Non-Clustered index is in one place and
refer to another place which stores data physically.
Because it need to refer back to the table,
Clustered index is slightly faster than a non clustered index.
-------
4.3.
A composite index is an index on two or more columns.
If you select ColmnA and ColumnB and
Both ColmnA and ColumnB are in the composite IndexA.
Then this is a covering query by the IndexA.
In this case, the data can simply be returned from the composite IndexA.
A Clustered Index always covers a query,
because it contains all data in a table.
*/
--=========================================================================
--T014_02_01
--By default, SQL server will set Primary Key as the clustered index
SELECT  *
FROM    Gamer;
/*
1.
Clustered Index
1.1.
One table can only have ONE clustered index.
By default, SQL server will set Primary Key as the clustered index
if there is no clustered index yet at that time.
1.2.
A Clustered index is stored with table and
does not need additional disk space.
it determines the storage order of data physically in the table.
1.3.
--SELECT  *
--FROM    Gamer;
It will order by Clustered index
*/
```

| | Id | FirstName | LastName | Email | Gender | GameScore |
|---|---|---|---|---|---|---|
| 1 | 1 | First1 | Last1 | 1@1.com | Male | 43000 |
| 2 | 2 | First2 | Last2 | 2@2.com | Female | 44000 |
| 3 | 3 | First3 | Last3 | 3@3.com | Female | 41000 |
| 4 | 4 | First4 | Last4 | 4@4.com | Male | 43000 |
| 5 | 5 | First5 | Last5 | 5@5.com | Male | 42000 |

```
--=========================================================================
--T014_02_02
--Create Clustered Index
```

```sql
CREATE CLUSTERED INDEX IX_Gamer_Name
ON Gamer(FirstName);
```

```
Messages
Msg 1902, Level 16, State 3, Line 285
Cannot create more than one clustered index on table 'Gamer'. Drop the existing clustered index 'PK__Gamer__3214EC0769346AED' before creating anothe
```

```sql
CREATE CLUSTERED INDEX IX_Gamer_Name_Gender
ON Gamer(FirstName DESC, Gender ASC);
```

```
Messages
Msg 1902, Level 16, State 3, Line 288
Cannot create more than one clustered index on table 'Gamer'. Drop the existing clustered index 'PK__Gamer__3214EC0769346AED' before creating anothe
```

```sql
GO -- Run the prvious command and begins new batch
/*
1.
--CREATE CLUSTERED INDEX IX_Gamer_Name
--ON Gamer(FirstName);
--CREATE CLUSTERED INDEX IX_Gamer_Name_Gender
--ON Gamer(FirstName DESC, Gender ASC);
Both Query will return Error
2.
One table can only have ONE clustered index.
By default, SQL server will set primary key Column as the clustered index
if there is no clustered index yet at that time.
*/
--=========================================================================
--T014_02_03
--sp_helpindex and Drop Clustered Index
EXECUTE sp_helpindex Gamer;
```

| | index_name | index_description | index_keys |
|---|---|---|---|
| 1 | PK__Gamer__3214EC0769346AED | clustered, unique, primary key located on PRIMARY | Id |

```sql
DROP INDEX Gamer.PK__Gamer__3214EC0732501013;
GO -- Run the prvious command and begins new batch
```

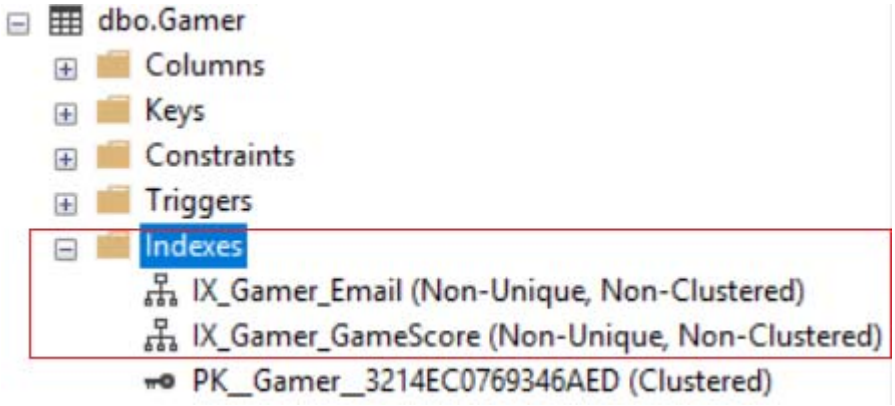
```
Messages
Msg 3701, Level 11, State 7, Line 311
Cannot drop the index 'Gamer.PK__Gamer__3214EC0732501013', because it does not exist or you do not have permission.
```

```sql
/*
1.
1.1.
--EXEC sp_helpindex N'TableName'
Get the Index of the Table
E.g.
EXEC sp_helpindex N'Gamer';
1.2.
--EXECUTE sp_helpindex Gamer;
You will this data row.
--index name :  PK__Gamer__3214EC071E222ACE
--index description  :  clustered, unique, primary key located on PRIMARY
--index_keys :  Id
2.
2.1.
--DROP INDEX Gamer.PK__Gamer__3214EC0732501013;
will return Error
2.2.
You can not use Query to drop Clustered Index
But you can drop Clustered Index in SSMS
In SSMS, delete the index
Database Name --> Table Name --> Indexes
--> Right Click --> Delete
Then we can delete the index in SSMS.
*/
```

==========

## 2.3. Nonclustered Index

```
--=========================================================================
--T014_03_Nonclustered Index
--=========================================================================
--=========================================================================
--T014_03_01
--Create Nonclustered Index
CREATE INDEX IX_Gamer_GameScore
ON Gamer (GameScore ASC);
CREATE NONCLUSTERED INDEX IX_Gamer_Email
ON Gamer(Email);
GO -- Run the prvious command and begins new batch
```




```
/*
1.
1.1.
Create Nonclustered Index Syntax1:
--CREATE INDEX IX_TableName_ColumnName
--ON TableName (ColumnName);
E.g.
--CREATE INDEX IX_Gamer_GameScore
--ON Gamer (GameScore ASC);
Create Nonclustered Index on GameScore
1.2.
Create Nonclustered Index Syntax2:
--CREATE NONCLUSTERED INDEX IX_TableName_ColumnName
--ON TableName (ColumnName);
E.g.
--CREATE NONCLUSTERED INDEX IX_Gamer_Email
--ON Gamer(Email);
Create Nonclustered Index on Email
*/
--=========================================================================
--T014_03_02
--sp_helpindex, Drop Index
EXEC sp_helpindex N'Gamer';
```

| | index_name | index_description | index_keys |
|---|---|---|---|
| 1 | IX_Gamer_Email | nonclustered located on PRIMARY | Email |
| 2 | IX_Gamer_GameScore | nonclustered located on PRIMARY | GameScore |
| 3 | PK__Gamer__3214EC0769346AED | clustered, unique, primary key located on PRIMARY | Id |

```
DROP INDEX Gamer.IX_Gamer_GameScore;
DROP INDEX Gamer.IX_Gamer_Email;
EXEC sp_helpindex N'Gamer';
```

```
GO -- Run the prvious command and begins new batch
```

| | index_name | index_description | index_keys |
|---|---|---|---|
| 1 | PK__Gamer__3214EC0769346AED | clustered, unique, primary key located on PRIMARY | Id |

```
/*
1.
--EXEC sp_helpindex N'TableName'
Get the Index of the Table
E.g.
EXEC sp_helpindex N'Gamer';
2.
SSMS - Create/Delete Index
2.1.
Create a new Index
Database Name --> Table Name --> Indexes
New Index --> Non-Clustered Index
Index Name
IX_Gamer_GameScore
-->
Add
--> Select the Column Name
--> OK
--> OK
2.2.
Delete/Disable an Index
Database Name --> Table Name --> Indexes
--> Right Click --> Delete or Disable
*/
```

## 2.4. T014_04_(Non)UniqueIndex_IGNORE_DUP_KEY

```
--==========================================================================
--T014_04_(Non)UniqueIndex_IGNORE_DUP_KEY
--==========================================================================
/*
1.
--DROP INDEX Gamer.PK__Gamer__3214EC0732501013;
will return Error
You can not use Query to drop Clustered Index
But you can drop Clustered Index in SSMS
In SSMS, delete the index
Database Name --> Table Name --> Indexes
--> Right Click --> Delete
Then we can delete the index in SSMS.
2.
UNIQUE is a property which can be assigned to
both CLUSTERED and NON-CLUSTERED indexes.
UNIQUE property ensure there are no duplicate data.
E.g.
If the index has UNIQUE property which
contains 2 columns, LastName and FirstName.
That means UNIQUE property ensures
there are no two enties has the same LastName and FirstName.
3.
By default of SQL server,
3.1.
When you create a PRIMARY KEY constraint
which automatically creates a unique clustered index.
3.2.
Add new UNIQUE CONSTRAINT will automatically add  UNIQUE NONCLUSTERED INDEX
Drop the  UNIQUE CONSTRAINT will automatically drop  UNIQUE NONCLUSTERED INDEX.
```

```
4.
if there are duplicate values in the Email column,
then you will have to do something
and ensure there is no duplicate values
before you set a UNIQUE constraint to Email Column.
5.
--CREATE UNIQUE INDEX IX_Gamer_DepartmentID
--ON Gamer(DepartmentID)
--WITH IGNORE_DUP_KEY;
when ColumnA have a unique index or constraint,
then ColumnA ensures there is no duplicate data.
E.g.
If I try to insert 5 data rows,
but there are 2 data rows contain duplicates.
Then all 5 data rows will be rejected.
--WITH IGNORE_DUP_KEY;
In this case, it allow to ignore thoese 2 duplicate rows.
and only insert the rest 3 data rows.
*/

--=========================================================================
--T014_04_01
--When you create Pimary Key, it automatically creates UNIQUE Index.
INSERT  INTO Gamer
VALUES  ( 1, 'First1A', 'Last1A', '1A@1.com', 'Male', 43000 );
```

Messages

```
Msg 2627, Level 14, State 1, Line 471
Violation of PRIMARY KEY constraint 'PK__Gamer__3214EC0769346AED'. Cannot insert duplicate key in object 'dbo.Gamer'. The d
The statement has been terminated.
```

```
/*
1.
--INSERT  INTO Gamer
--VALUES  ( 1, 'First1A', 'Last1A', '1A@1.com', 'Male', 43000 );
will return Error
When you create Pimary Key, it automatically creates UNIQUE Index.
Thus, you may not enter duplicate data in Primary Key.
2.
UNIQUE is a property which can be assigned to
both CLUSTERED and NON-CLUSTERED indexes.
UNIQUE property ensure there are no duplicate data.
E.g.
If the index has UNIQUE property which
contains 2 columns, LastName and FirstName.
That means UNIQUE property ensures
there are no two enties has the same LastName and FirstName.
3.
By default of SQL server,
3.1.
When you create a PRIMARY KEY constraint
which automatically creates a unique clustered index.
3.2.
Add new UNIQUE CONSTRAINT will automatically add  UNIQUE NONCLUSTERED INDEX
Drop the  UNIQUE CONSTRAINT will automatically drop  UNIQUE NONCLUSTERED INDEX.
*/
--=========================================================================
--T014_04_02
--sp_helpindex and Drop Clustered Index
EXECUTE sp_helpindex Gamer;
```

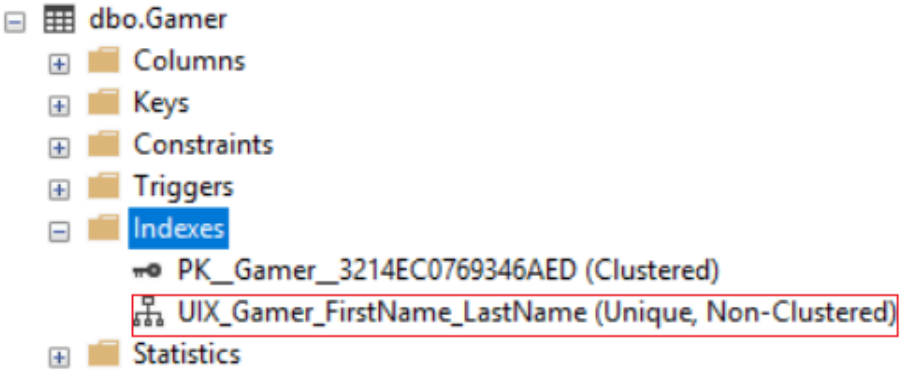
| | index_name | index_description | index_keys |
|---|---|---|---|
| 1 | PK__Gamer__3214EC0769346AED | clustered, unique, primary key located on PRIMARY | Id |

```
DROP INDEX Gamer.PK__Gamer__3214EC0732501013;
GO -- Run the prvious command and begins new batch
```

```
Messages
Msg 3701, Level 11, State 7, Line 506
Cannot drop the index 'Gamer.PK__Gamer__3214EC0732501013', because it does not exist or you do not have permission.
```

```
/*
1.
1.1.
--EXEC sp_helpindex N'TableName'
Get the Index of the Table
E.g.
EXEC sp_helpindex N'Gamer';
1.2.
--EXECUTE sp_helpindex Gamer;
You will this data row.
--index name :  PK__Gamer__3214EC071E222ACE
--index description  :  clustered, unique, primary key located on PRIMARY
--index_keys :  Id
2.
2.1.
--DROP INDEX Gamer.PK__Gamer__3214EC0732501013;
will return Error
2.2.
You can not use Query to drop Clustered Index
But you can drop Clustered Index in SSMS
In SSMS, delete the index
Database Name --> Table Name --> Indexes
--> Right Click --> Delete
Then we can delete the index in SSMS.
*/

--==========================================================================
--T014_04_03
--Add new UNIQUE NONCLUSTERED INDEX
CREATE UNIQUE NONCLUSTERED INDEX UIX_Gamer_FirstName_LastName
ON Gamer(FirstName, LastName);
```



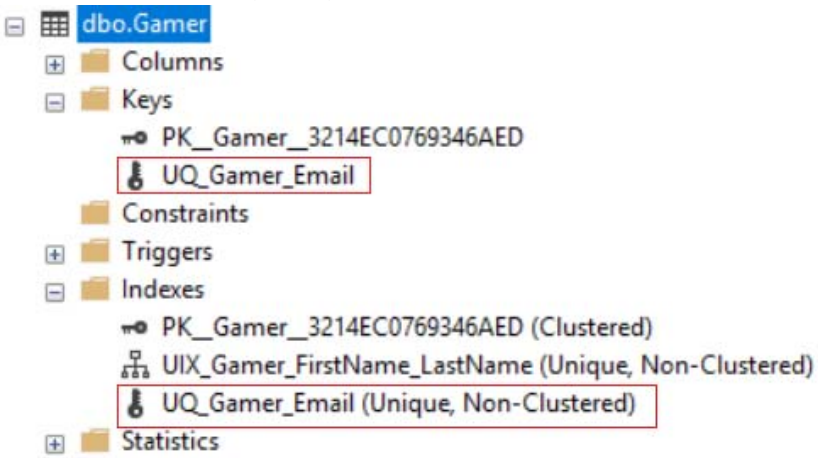

```
INSERT  INTO Gamer
VALUES  ( 6, 'First1', 'Last1', '1@1.com', 'Male', 43000 );
--Return Error
```

```
Messages
Msg 2601, Level 14, State 1, Line 542
Cannot insert duplicate key row in object 'dbo.Gamer' with unique index 'UIX_Gamer_FirstName_LastName'. The duplicate key value is (Firs
The statement has been terminated.
```

```
INSERT  INTO Gamer
VALUES  ( 6, 'First6', 'Last1', '6@6.com', 'Male', 43000 );
--Insert Successfully
SELECT  *
FROM    Gamer;
GO -- Run the prvious command and begins new batch
```

| | Id | FirstName | LastName | Email | Gender | GameScore |
|---|---|---|---|---|---|---|
| 1 | 1 | First1 | Last1 | 1@1.com | Male | 43000 |
| 2 | 2 | First2 | Last2 | 2@2.com | Female | 44000 |
| 3 | 3 | First3 | Last3 | 3@3.com | Female | 41000 |
| 4 | 4 | First4 | Last4 | 4@4.com | Male | 43000 |
| 5 | 5 | First5 | Last5 | 5@5.com | Male | 42000 |
| 6 | 6 | First6 | Last1 | 6@6.com | Male | 43000 |

```
/*
1.
--CREATE UNIQUE NONCLUSTERED INDEX UIX_Gamer_FirstName_LastName
--ON Gamer(FirstName, LastName);
Create an UNIQUE NONCLUSTERED INDEX for FirstName and LastName columns.
--INSERT  INTO Gamer
--VALUES  ( 6, 'First1', 'Last1', '1@1.com', 'Male', 43000 );
'First1', 'Last1' is already existed, thus insert will fail.
--INSERT  INTO Gamer
--VALUES  ( 6, 'First6', 'Last1', '6@6.com', 'Male', 43000 );
'First6', 'Last1' is Not existed, thus insert will be success.
2.
UNIQUE is a property which can be assigned to
both CLUSTERED and NON-CLUSTERED indexes.
UNIQUE property ensure there are no duplicate data.
E.g.
If the index has UNIQUE property which
contains 2 columns, LastName and FirstName.
That means UNIQUE property ensures
there are no two enties has the same LastName and FirstName.
*/
--=========================================================================
--T014_04_04
--Add new UNIQUE CONSTRAINT, automatically add  UNIQUE NONCLUSTERED INDEX
ALTER TABLE Gamer
ADD CONSTRAINT UQ_Gamer_Email
UNIQUE NONCLUSTERED (Email);
```




```
INSERT  INTO Gamer
VALUES  ( 7, 'First7', 'Last7', '1@1.com', 'Male', 43000 );
--Return Error
```


```
INSERT  INTO Gamer
VALUES  ( 7, 'First7', 'Last7', '7@7.com', 'Male', 43000 );
```

```
--Insert Successfully

SELECT  *
FROM    Gamer;
GO -- Run the prvious command and begins new batch
```

| | Id | FirstName | LastName | Email | Gender | GameScore |
|---|---|---|---|---|---|---|
| 1 | 1 | First1 | Last1 | 1@1.com | Male | 43000 |
| 2 | 2 | First2 | Last2 | 2@2.com | Female | 44000 |
| 3 | 3 | First3 | Last3 | 3@3.com | Female | 41000 |
| 4 | 4 | First4 | Last4 | 4@4.com | Male | 43000 |
| 5 | 5 | First5 | Last5 | 5@5.com | Male | 42000 |
| 6 | 6 | First6 | Last1 | 6@6.com | Male | 43000 |
| 7 | 7 | First7 | Last7 | 7@7.com | Male | 43000 |

```
/*
1.
--ALTER TABLE Gamer
--ADD CONSTRAINT UQ_Gamer_Email
--UNIQUE NONCLUSTERED (Email);
Add new UNIQUE CONSTRAINT, automatically add  UNIQUE NONCLUSTERED INDEX
--INSERT  INTO Gamer
--VALUES  ( 7, 'First7', 'Last7', '1@1.com', 'Male', 43000 );
'1@1.com' is already existed, thus insert will fail.
--INSERT  INTO Gamer
--VALUES  ( 7, 'First7', 'Last7', '7@7.com', 'Male', 43000 );
'7@7.com' is Not existed, thus insert will be success.
2.
Add new UNIQUE CONSTRAINT will automatically add  UNIQUE NONCLUSTERED INDEX
Drop the  UNIQUE CONSTRAINT will automatically drop  UNIQUE NONCLUSTERED INDEX.
*/

--=========================================================================
--T014_04_05
--sys.sp_helpconstraint   V.S.   sys.sp_helpindex
EXECUTE sys.sp_helpconstraint @objname = N'Gamer';
```

| | Object Name |
|---|---|
| 1 | Gamer |

| | constraint_type | constraint_name | delete_action | update_action | status_enabled | status_for_replication | constraint_keys |
|---|---|---|---|---|---|---|---|
| 1 | PRIMARY KEY (clustered) | PK__Gamer__3214EC0769346AED | (n/a) | (n/a) | (n/a) | (n/a) | Id |
| 2 | UNIQUE (non-clustered) | UQ_Gamer_Email | (n/a) | (n/a) | (n/a) | (n/a) | Email |

```
EXECUTE sys.sp_helpindex @objname = N'Gamer';
```

| | index_name | index_description | index_keys |
|---|---|---|---|
| 1 | PK__Gamer__3214EC0769346AED | clustered, unique, primary key located on PRIMARY | Id |
| 2 | UIX_Gamer_FirstName_LastName | nonclustered, unique located on PRIMARY | FirstName, LastName |
| 3 | UQ_Gamer_Email | nonclustered, unique, unique key located on PRIMA... | Email |

```
ALTER TABLE Gamer
DROP CONSTRAINT UQ_Gamer_Email;
EXECUTE sys.sp_helpconstraint @objname = N'Gamer';
```

| | Object Name |
|---|---|
| 1 | Gamer |

| | constraint_type | constraint_name | delete_action | update_action | status_enabled | status_for_replication | constraint_keys |
|---|---|---|---|---|---|---|---|
| 1 | PRIMARY KEY (clustered) | PK__Gamer__3214EC0769346AED | (n/a) | (n/a) | (n/a) | (n/a) | Id |

```
EXECUTE sys.sp_helpindex @objname = N'Gamer';
```

| | index_name | index_description | index_keys |
|---|---|---|---|
| 1 | PK__Gamer__3214EC0769346AED | clustered, unique, primary key located on PRIMARY | Id |
| 2 | UIX_Gamer_FirstName_LastName | nonclustered, unique located on PRIMARY | FirstName, LastName |

```
GO -- Run the prvious command and begins new batch
/*
1.
--sys.sp_helpconstraint   V.S.   sys.sp_helpindex
1.1.
--EXECUTE sys.sp_helpconstraint @objname = N'TableName';
Get the constraint information of the Table
1.2.
--EXECUTE sys.sp_helpindex @objname = N'TableName';
Or
--EXEC sp_helpindex N'TableName'
Get the Index information of the Table
E.g.
EXEC sp_helpindex N'Gamer';
2.
Add new UNIQUE CONSTRAINT will automatically add  UNIQUE NONCLUSTERED INDEX
Drop the  UNIQUE CONSTRAINT will automatically drop  UNIQUE NONCLUSTERED INDEX.
*/
--=========================================================================
--T014_04_06
--WITH IGNORE_DUP_KEY;
SELECT  *
FROM    Gamer;
```

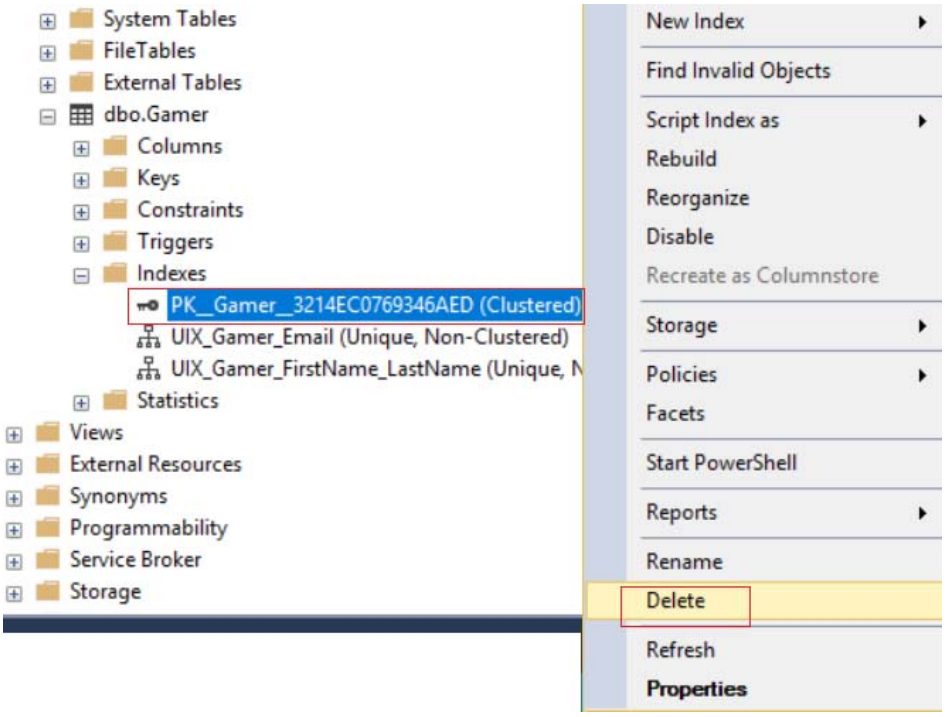
| | Id | FirstName | LastName | Email | Gender | GameScore |
|---|---|---|---|---|---|---|
| 1 | 1 | First1 | Last1 | 1@1.com | Male | 43000 |
| 2 | 2 | First2 | Last2 | 2@2.com | Female | 44000 |
| 3 | 3 | First3 | Last3 | 3@3.com | Female | 41000 |
| 4 | 4 | First4 | Last4 | 4@4.com | Male | 43000 |
| 5 | 5 | First5 | Last5 | 5@5.com | Male | 42000 |
| 6 | 6 | First6 | Last1 | 6@6.com | Male | 43000 |
| 7 | 7 | First7 | Last7 | 7@7.com | Male | 43000 |

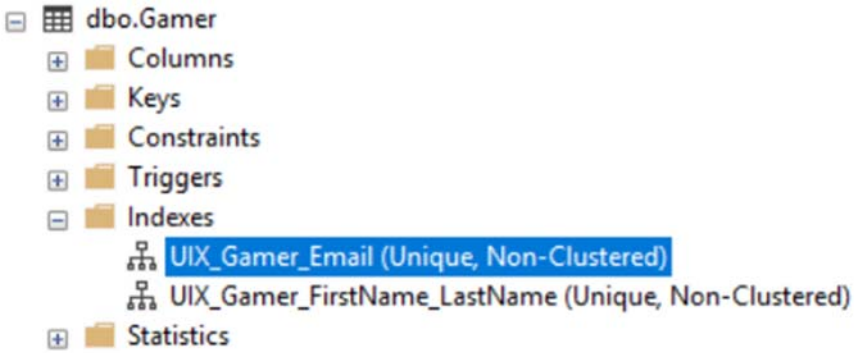
```
--Create UIX_Gamer_Email WITH IGNORE_DUP_KEY;
CREATE UNIQUE INDEX UIX_Gamer_Email
ON Gamer(Email)
WITH IGNORE_DUP_KEY;
--Insert 2 rows with Duplicate Email, these 2 rows insert will fail.
INSERT  INTO Gamer
VALUES  ( 8, 'First8', 'Last8', '1@1.com', 'Female', 44000 );
INSERT  INTO Gamer
VALUES  ( 9, 'First9', 'Last9', '2@2.com', 'Male', 43000 );
--Insert 3 rows WITHOUT Duplicate Email, these 2 rows insert will success.
INSERT  INTO Gamer
VALUES  ( 10, 'First10', 'Last10', '10@10.com', 'Female', 44000 );
INSERT  INTO Gamer
VALUES  ( 11, 'First11', 'Last11', '11@11.com', 'Male', 43000 );
INSERT  INTO Gamer
VALUES  ( 12, 'First12', 'Last12', '12@12.com', 'Male', 43000 );
SELECT  *
FROM    dbo.Gamer;
GO -- Run the prvious command and begins new batch
```

| | Id | FirstName | LastName | Email | Gender | GameScore |
|---|---|---|---|---|---|---|
| 1 | 1 | First1 | Last1 | 1@1.com | Male | 43000 |
| 2 | 2 | First2 | Last2 | 2@2.com | Female | 44000 |
| 3 | 3 | First3 | Last3 | 3@3.com | Female | 41000 |
| 4 | 4 | First4 | Last4 | 4@4.com | Male | 43000 |
| 5 | 5 | First5 | Last5 | 5@5.com | Male | 42000 |
| 6 | 6 | First6 | Last1 | 6@6.com | Male | 43000 |
| 7 | 7 | First7 | Last7 | 7@7.com | Male | 43000 |
| 8 | 10 | First10 | Last10 | 10@10.com | Female | 44000 |
| 9 | 11 | First11 | Last11 | 11@11.com | Male | 43000 |
| 10 | 12 | First12 | Last12 | 12@12.com | Male | 43000 |

```
/*
1.
Output as following
--Duplicate key was ignored.
--(0 rows affected)
--Duplicate key was ignored.
--(0 rows affected)
--(1 row affected)
--(1 row affected)
--(1 row affected)
2.
--CREATE UNIQUE INDEX UIX_Gamer_Email
--ON Gamer(Email)
--WITH IGNORE_DUP_KEY;
when ColumnA have a unique index or constraint,
then ColumnA ensures there is no duplicate data.
E.g.
If I try to insert 5 data rows,
but there are 2 data rows contain duplicates.
Then all 5 data rows will be rejected.
--WITH IGNORE_DUP_KEY;
In this case, it allow to ignore thoese 2 duplicate rows.
and only insert the rest 3 data rows.
*/

--=========================================================================
--T014_04_07
--Delete INDEX Gamer.PK__Gamer__3214EC0732501013 in SSMS and insert duplicate data.
```




```sql
SELECT  *
FROM    dbo.Gamer;
```

| | Id | FirstName | LastName | Email | Gender | GameScore |
|---|---|---|---|---|---|---|
| 1 | 1 | First1 | Last1 | 1@1.com | Male | 43000 |
| 2 | 2 | First2 | Last2 | 2@2.com | Female | 44000 |
| 3 | 3 | First3 | Last3 | 3@3.com | Female | 41000 |
| 4 | 4 | First4 | Last4 | 4@4.com | Male | 43000 |
| 5 | 5 | First5 | Last5 | 5@5.com | Male | 42000 |
| 6 | 6 | First6 | Last1 | 6@6.com | Male | 43000 |
| 7 | 7 | First7 | Last7 | 7@7.com | Male | 43000 |
| 8 | 10 | First10 | Last10 | 10@10.com | Female | 44000 |
| 9 | 11 | First11 | Last11 | 11@11.com | Male | 43000 |
| 10 | 12 | First12 | Last12 | 12@12.com | Male | 43000 |

```sql
INSERT  INTO Gamer
VALUES  ( 1, 'First1A', 'Last1A', '1A@1.com', 'Male', 43000 );
INSERT  INTO Gamer
VALUES  ( 1, 'First1B', 'Last1B', '1B@1.com', 'Male', 43000 );
SELECT  *
FROM    dbo.Gamer;
GO -- Run the prvious command and begins new batch
```

| | Id | FirstName | LastName | Email | Gender | GameScore |
|---|---|---|---|---|---|---|
| 1 | 1 | First1 | Last1 | 1@1.com | Male | 43000 |
| 2 | 2 | First2 | Last2 | 2@2.com | Female | 44000 |
| 3 | 3 | First3 | Last3 | 3@3.com | Female | 41000 |
| 4 | 4 | First4 | Last4 | 4@4.com | Male | 43000 |
| 5 | 5 | First5 | Last5 | 5@5.com | Male | 42000 |
| 6 | 6 | First6 | Last1 | 6@6.com | Male | 43000 |
| 7 | 7 | First7 | Last7 | 7@7.com | Male | 43000 |
| 8 | 10 | First10 | Last10 | 10@10.com | Female | 44000 |
| 9 | 11 | First11 | Last11 | 11@11.com | Male | 43000 |
| 10 | 12 | First12 | Last12 | 12@12.com | Male | 43000 |
| 11 | 1 | First1A | Last1A | 1A@1.com | Male | 43000 |
| 12 | 1 | First1B | Last1B | 1B@1.com | Male | 43000 |

```
/*
1.
--INSERT  INTO Gamer
--VALUES  ( 1, 'First1A', 'Last1A', '1A@1.com', 'Male', 43000 );
will return Error
When you create Pimary Key, it automatically creates UNIQUE Index.
Thus, you may not enter duplicate data in Primary Key.
2.
Now,
Delete INDEX Gamer.PK__Gamer__3214EC0732501013 in SSMS and insert duplicate data.
and insert duplicated Id again.
--INSERT  INTO Gamer
--VALUES  ( 1, 'First1A', 'Last1A', '1A@1.com', 'Male', 43000 );
--INSERT  INTO Gamer
--VALUES  ( 1, 'First1B', 'Last1B', '1B@1.com', 'Male', 43000 );
Now, both rows will be inserted successfully.
*/
```

## 2.5. T014_05_GoodAndBadOfIndexes

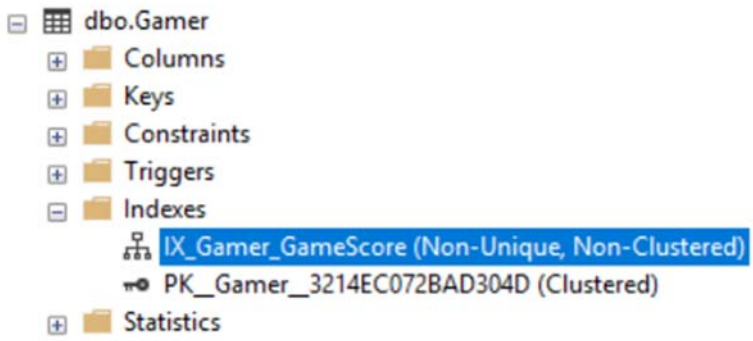
```
--=========================================================================
--T014_05_GoodAndBadOfIndexes
--=========================================================================
/*
5.
5.1.
Good at Index:
If we have Index in ColumnA,
Index is good for
WHERE, WHERE with DELETE/UPDATE, ORDER BY, GROUP BY in ColumnA
5.2.
Bad at Index:
5.2.1.
NonClustered Index need additional disk space.
5.2.2.
When there are a lof of data in the table,
then DELETE or UPDATE performace might be bad.
Because it need extra time to update Indexes.
*/

--=========================================================================
--T014_05_01
--Create Sample Data
```

```sql
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE      TABLE_NAME = 'Gamer' ) )
    BEGIN
        TRUNCATE TABLE Gamer;
        DROP TABLE Gamer;
    END;
GO -- Run the previous command and begins new batch
CREATE TABLE Gamer
(
  Id INT PRIMARY KEY ,
  FirstName NVARCHAR(100) ,
  LastName NVARCHAR(100) ,
  Email NVARCHAR(100) ,
  Gender NVARCHAR(10) ,
  GameScore INT
);
GO -- Run the previous command and begins new batch
INSERT  INTO Gamer
VALUES  ( 4, 'First4', 'Last4', '4@4.com', 'Male', 43000 );
INSERT  INTO Gamer
VALUES  ( 2, 'First2', 'Last2', '2@2.com', 'Female', 44000 );
INSERT  INTO Gamer
VALUES  ( 1, 'First1', 'Last1', '1@1.com', 'Male', 43000 );
INSERT  INTO Gamer
VALUES  ( 5, 'First5', 'Last5', '5@5.com', 'Male', 42000 );
INSERT  INTO Gamer
VALUES  ( 3, 'First3', 'Last3', '3@3.com', 'Female', 41000 );
GO -- Run the previous command and begins new batch
SELECT  *
FROM     Gamer;
GO -- Run the previous command and begins new batch
```

| | Id | FirstName | LastName | Email | Gender | GameScore |
|---|---|---|---|---|---|---|
| 1 | 1 | First1 | Last1 | 1@1.com | Male | 43000 |
| 2 | 2 | First2 | Last2 | 2@2.com | Female | 44000 |
| 3 | 3 | First3 | Last3 | 3@3.com | Female | 41000 |
| 4 | 4 | First4 | Last4 | 4@4.com | Male | 43000 |
| 5 | 5 | First5 | Last5 | 5@5.com | Male | 42000 |

```sql
--=====================================================================
--T014_05_02
--Create a Non-Clustered Index on GameScore Column
CREATE NONCLUSTERED INDEX IX_Gamer_GameScore
ON Gamer(GameScore ASC);
GO -- Run the previous command and begins new batch
```




```
--=========================================================================
--T014_05_03
--Index is good for WHERE
SELECT  *
FROM    dbo.Gamer
WHERE   GameScore > 41000
        AND GameScore < 48000;
GO -- Run the previous command and begins new batch
```

| | Id | FirstName | LastName | Email | Gender | GameScore |
|---|---|---|---|---|---|---|
| 1 | 1 | First1 | Last1 | 1@1.com | Male | 43000 |
| 2 | 2 | First2 | Last2 | 2@2.com | Female | 44000 |
| 3 | 4 | First4 | Last4 | 4@4.com | Male | 43000 |
| 4 | 5 | First5 | Last5 | 5@5.com | Male | 42000 |

```
/*
Because GameScore Column has a Non-Clustered Index.
Thus, SQL server doesn't have to search from first row to last row.
SQL server will just look at index and
find out the exact address of the data.
*/
--=========================================================================
--T014_05_04
--Index is good for WHERE with Delete or Update
SELECT  *
FROM    dbo.Gamer



DELETE  FROM Gamer
WHERE   GameScore = 44000;
UPDATE  dbo.Gamer
SET     GameScore = 49000
WHERE   GameScore = 41000;
SELECT  *
FROM    dbo.Gamer
GO -- Run the previous command and begins new batch



/*
Because GameScore Column has a Non-Clustered Index.
Thus, SQL server doesn't have to search from first row to last row.
SQL server will just look at index and
find out the exact address of the data.
Then Update or Delete it
```

```
*/
--=========================================================================
--T014_05_05
--Index is good for ORDER BY
SELECT  *
FROM    dbo.Gamer
ORDER BY GameScore;
SELECT  *
FROM    Gamer
ORDER BY GameScore DESC;
GO -- Run the previous command and begins new batch




/*
Because GameScore Column has a Non-Clustered Index.
Thus, SQL server doesn't have to search from first row to last row.
SQL server will just look at index, then order the rows by using index.
*/
--=========================================================================
--T014_05_06
--Index is good for GROUP BY
SELECT  GameScore ,
        COUNT(GameScore) AS TotalGameScore
FROM    dbo.Gamer
GROUP BY GameScore;




/*
Because GameScore Column has a Non-Clustered Index.
Thus, SQL server doesn't have to search from first row to last row.
SQL server will just look at index, then group the rows by using index.
*/
```

## 2.5. T014_06_GoodAndBadOfIndexes

```
--=========================================================================
--T014_06_GoodAndBadOfIndexes
--=========================================================================
--Clean up
IF ( EXISTS ( SELECT    *
              FROM      INFORMATION_SCHEMA.TABLES
              WHERE      TABLE_NAME = 'Gamer' ) )
    BEGIN
        TRUNCATE TABLE Gamer;
        DROP TABLE Gamer;
    END;
GO -- Run the previous command and begins new batch
```