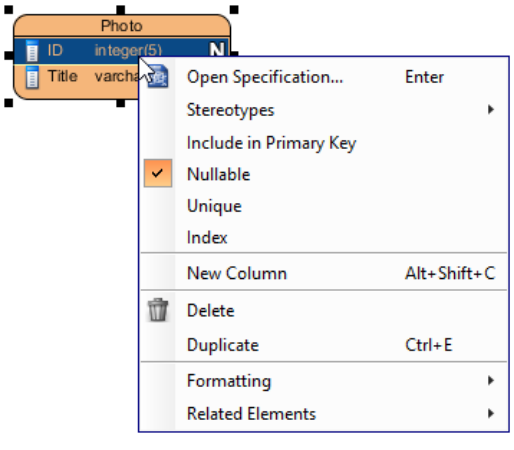
Column in Detail

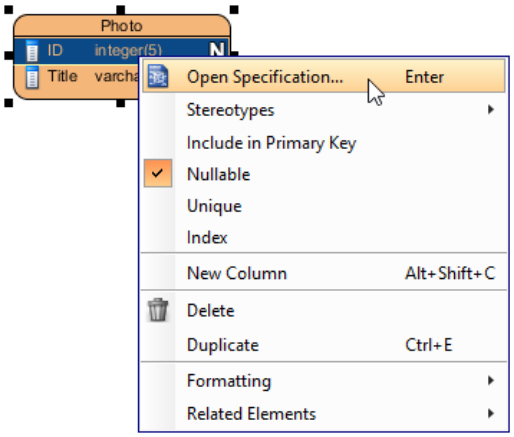
Opening column specification

1. Right click on a column.



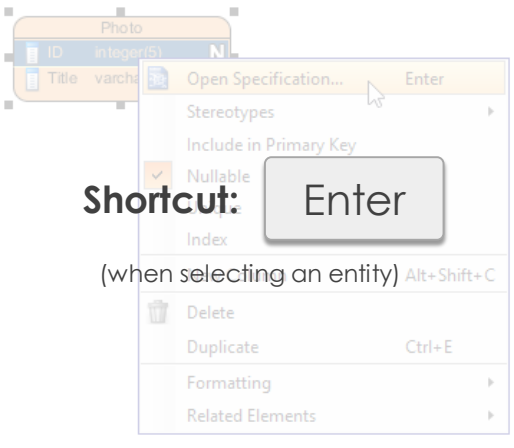
Opening column specification

1. Right click on a column.
2. Select **Open Specification…** from the popup menu.

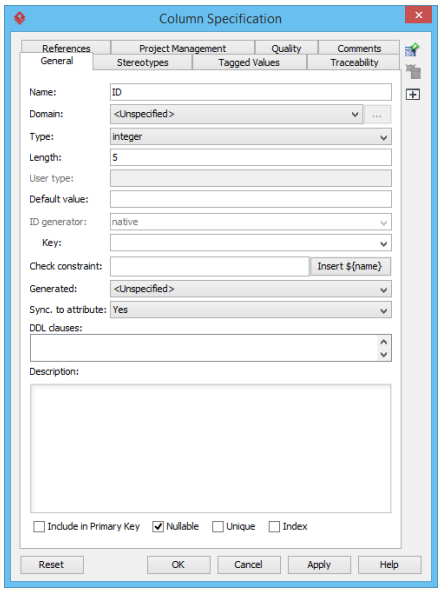


Opening column specification

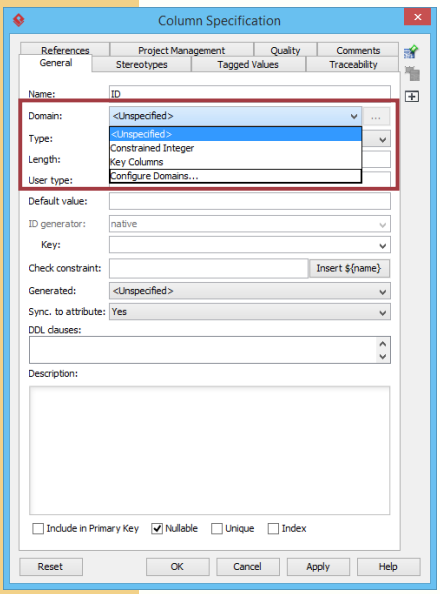
1. Right click on a column.
2. Select **Open Specification…** from the popup menu.



Opening column specification

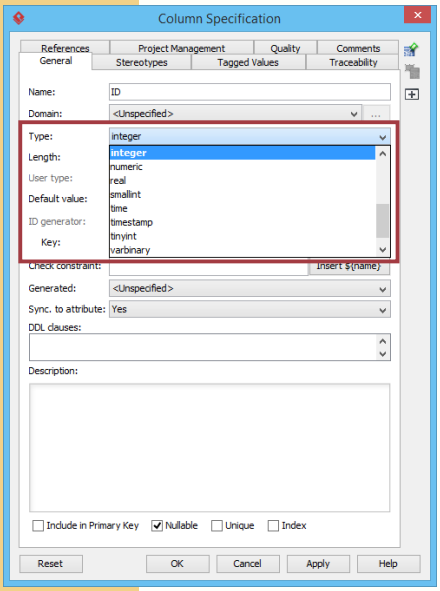


Column details ■ Domain



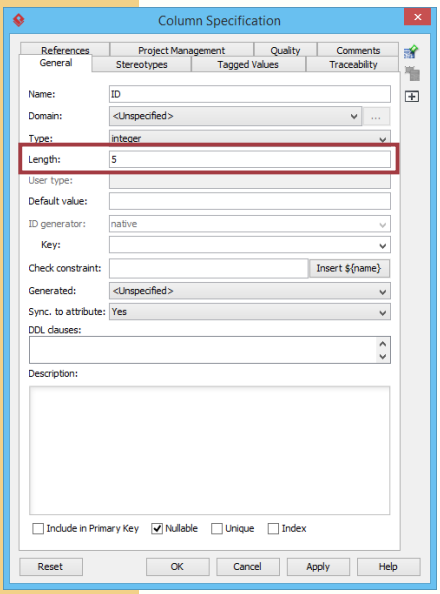
- Definition: A named set of property values that can be re-used by a column.
- Usage
  1. Create a domain for commonly entered values such as type, length and default value.
  2. Re-use it in other columns when needed.
- Benefits
  - Save time in creating **similar** columns
  - Keep data model consistent

Column details ■ Type



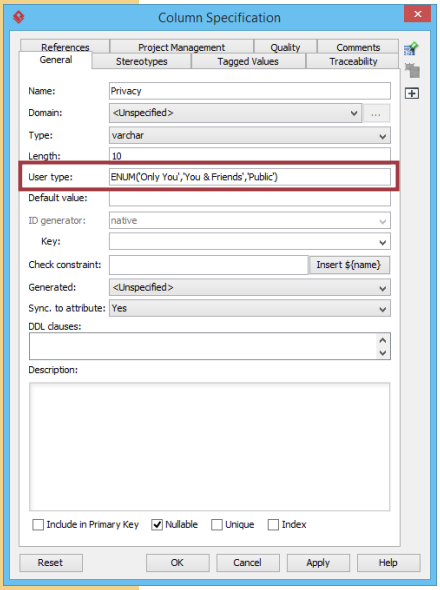
- Identify the type of data allowed to be stored in the column.
  - E.g. integer, char, varchar, long, float, datetime
- Different vendors may support the same type differently
  - E.g.
    - decimal in MySQL, float in Oracle
    - timestamp in Oracle, datetime in MSSQL
- You can configure the DBMS you use in Database Configuration (**Tools > DB > Database Configuration**)

Column details ■ Length



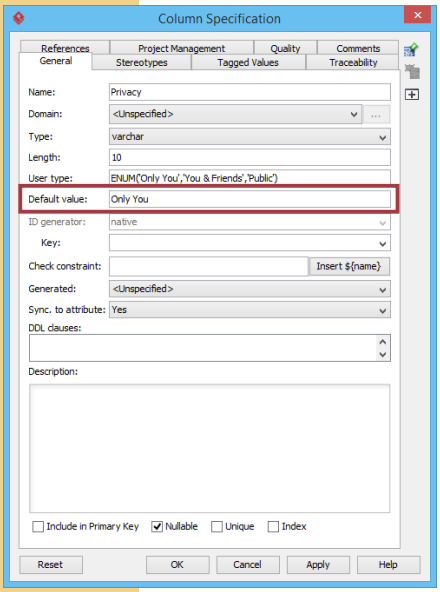
- The maximum allowable length of data that can be stored under the column, per each record.

Column details ■ User Type



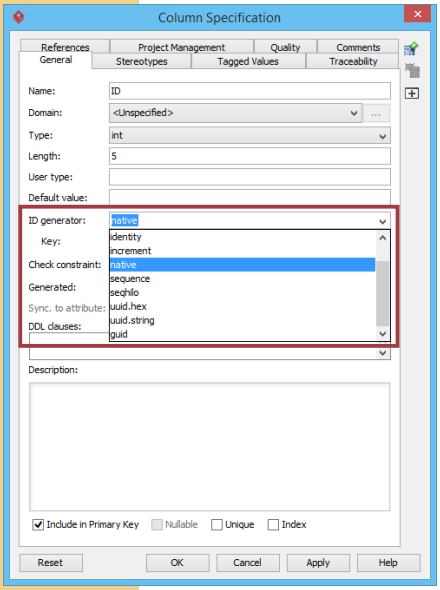
- Define enumeration values
- Effective in database generation

Column details ■ Default Value



- A default value of the column when no value is specified when inserting data into the database.

Column details ■ ID Generator

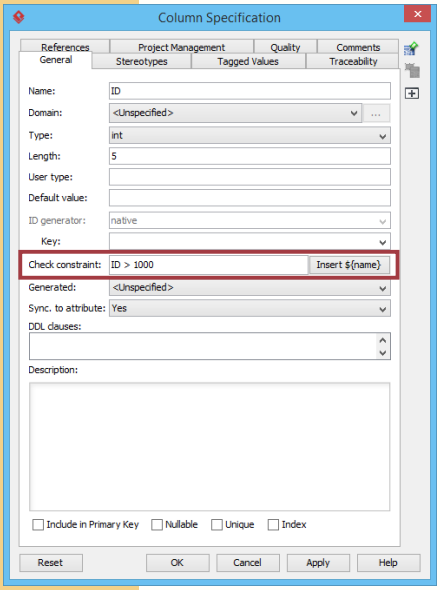


- The strategy of how unique values will be produced in a **primary key** column.

Column details ■ ID Generator

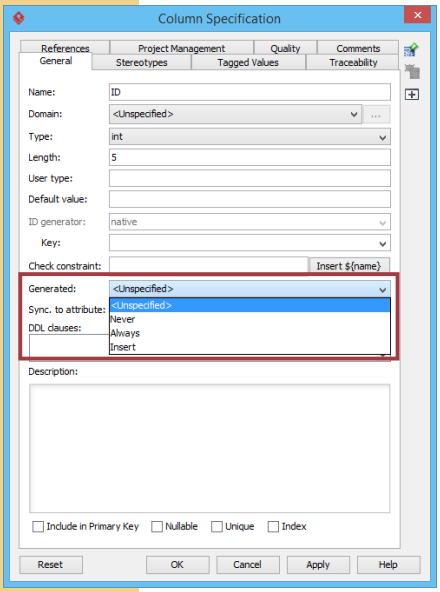
ID Generator	Description
assigned	allows the application to assign an identifier to the object before save() is called.
guid	uses a database-generated GUID string on MS SQL Server and MySQL.
hilo	uses a hi/lo algorithm to efficiently generate identifiers of type long, short or int, given a table and column as a source of hi values. The hi/lo algorithm generates identifiers that are unique only for a particular database.
identity	supports identity columns in DB2, MySQL, MS SQL Server, Sybase and HypersonicSQL. The returned identifier is of type long, short or int.
increment	generates identifiers of type long, short or int that are unique only when no other process is inserting data into the same table. Do not use in a cluster.
native	(default) picks identity, sequence or hilo depending upon the capabilities of the underlying database.
seqhilo	uses a hi/lo algorithm to efficiently generate identifiers of type long, short or int, given a named database sequence.
Sequence	uses a sequence in DB2, PostgreSQL, Oracle. The returned identifier is of type long,short or int

Column details ■ Check Constraint



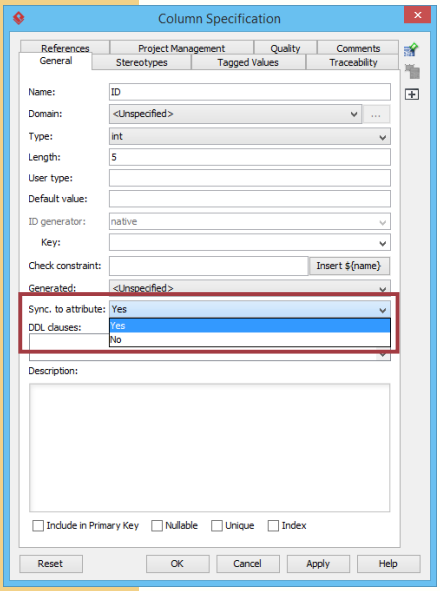
- Allows only certain values to be filled in a column.

Column details ■ Generated



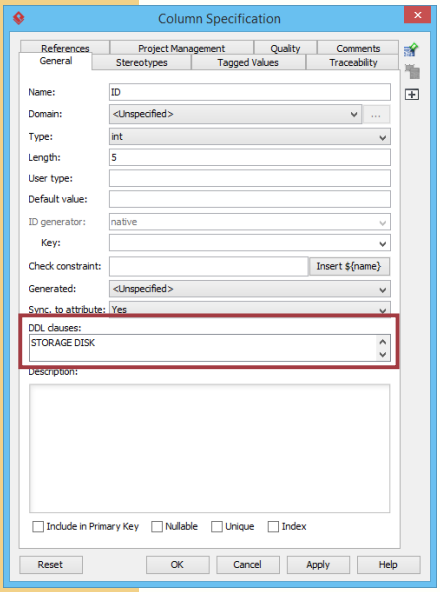
- Used for ORM. It marks the column to be generated when insert, or both insert/update
  - e.g. a trigger will update column "lastModified" by current time when insert or update. When generated = always, Hibernate will execute "select lastModified ..." after insert/update to update the ORM model.

Column details ■ Sync. to Attribute



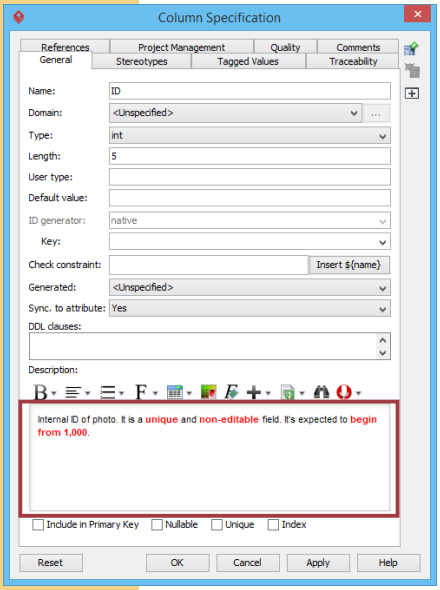
- Enable/disable synchronizing the column with the attribute in the corresponding class
  - Column name <-> Attribute name
  - Column type <-> Attribute data type

Column details ■ DDL Clauses



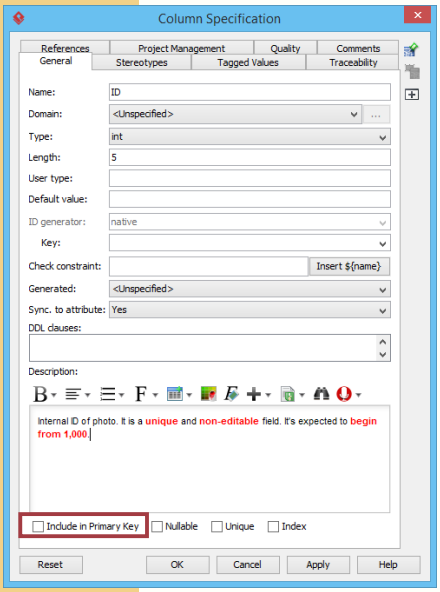
- The text to include in the create table statement in database generation

Column details ■ Description



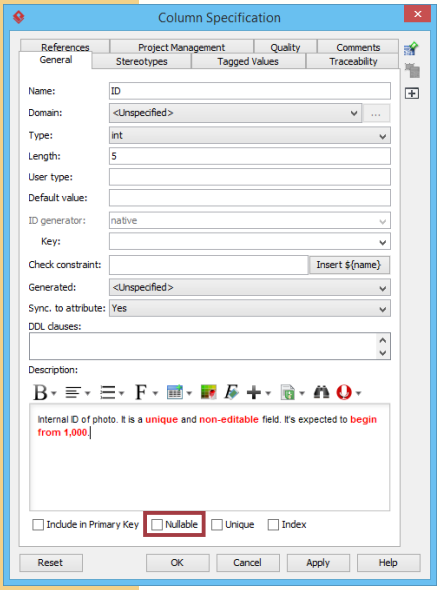
- Describe the column in detail
- Useful in:
  - Creating data dictionary
  - Documenting database table (as comment of columns)

Column details ■ Include in Primary Key



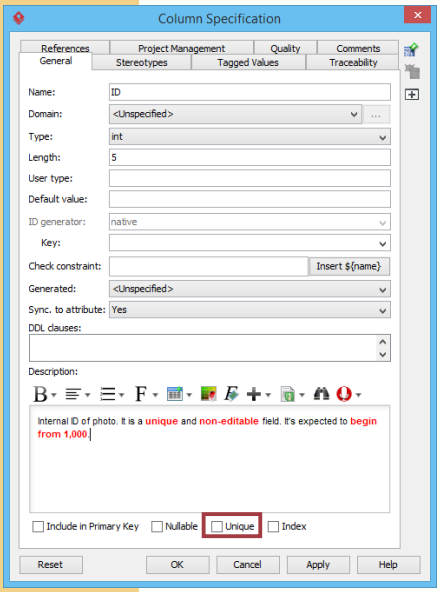
- A primary key (PK) uniquely identifies each record in a table.
- One entity can have zero or one primary key.
- One primary key can be formed by multiple columns.

Column details ■ Nullable



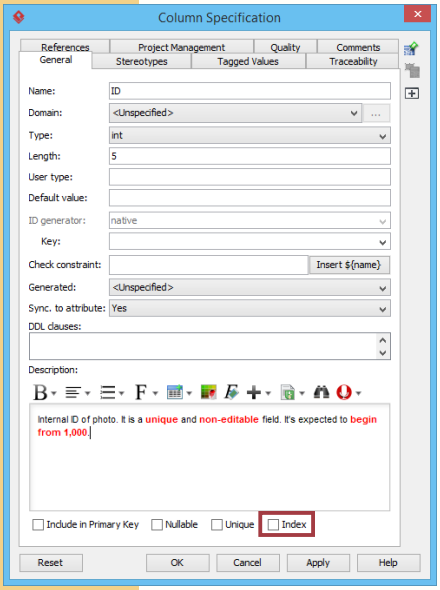
- Allow value of the column to be set as NULL
- Situations when applicable:
  - Value is to be filled later
  - Value can be unspecified
- Make sure the use of nullable field does not break your business rule

Column details ■ Unique

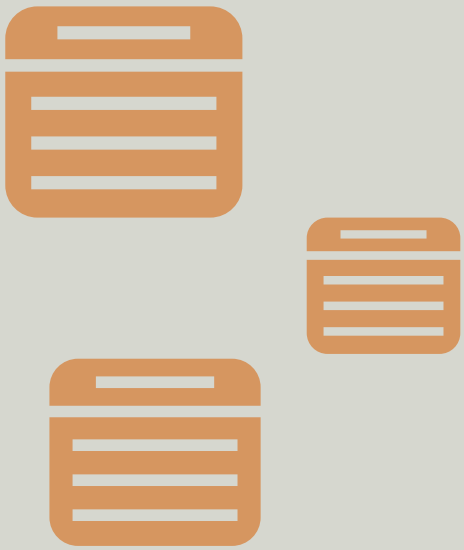


- Used to ensure data is not duplicated in any two rows in a database table.
- Can be null

Column details ■ Index



- Index improves the speed of data retrieval on a table.
- You set a column to be the index column when data retrieval has to be made frequently through that column.



Index (Entity)

Querying data without index

Uploader Email	Uploader Name	Uploader Address
donald.t.lopresti@gmail.com	Donald T. Lopresti	572 Lost Lake Crossing
michele.r.herrmann@gmail.com	Michele R. Herrmann	213 White Swan Garden
wa		asse
car		eet
kim		ue
tony.stanley.ca@gmail.com	Tony Stanley	2823 Percy Avenue
heleb.b.broom@gmail.com	Helen Broom	746 Happy Valley

Let's find him!

tony.stanley.ca@gmail.com

Column in Detail

Querying data without index

Uploader Email	Uploader Name	Uploader Address
donald.t.lopresti@gmail.com	Donald T. Lopresti	572 Lost Lake Crossing
michele.r.herrmann@gmail.com	Michele R. Herrmann	213 White Swan Garden
wanda.a.Allison@gmail.com	Wanda A. Allison	2017 Green Impasse
carol.faulkner.us@gmail.com	Carol Faulkner	2841 Rodney Street
kimkim.g.1980@gmail.com	Kimberly Greenlee	2314 Pearl Avenue
tony.stanley.ca@gmail.com	Tony Stanley	2823 Percy Avenue
heleb.b.broom@gmail.com	Helen Broom	746 Happy Valley

Column in Detail

Index speeds up search queries by

reducing the number of records that need to be examined

Querying data

Index Column

Uploader Email	Uploader Name	Uploader Address
donald.t.lopresti@gmail.com	Donald T. Lopresti	572 Lost Lake Crossing
michele.r.herrmann@gmail.com	Michele R. Herrmann	213 White Swan Garden
wanda.a.Allison@gmail.com	Wanda A. Allison	2017 Green Impasse
carol.faulkner.us@gmail.com	Carol Faulkner	2841 Rodney Street
kimkim.g.1980@gmail.com	Kimberly Greenlee	2314 Pearl Avenue
⋮	⋮	⋮
tony.stanley.ca@gmail.com	Tony Stanley	2823 Percy Avenue
heleb.b.broom@gmail.com	Helen Broom	746 Happy Valley

Querying data with index

Uploader Email	Uploader Name	Uploader Address
donald.t.lopresti@gmail.com	Donald T. Lopresti	572 Lost Lake Crossing
michele.r.herrmann@gmail.com	Michele R. Herrmann	213 White Swan Garden
wanda.a.Allison@gmail.com	Wanda A. Allison	2017 Green Impasse
carol.faulkner.us@gmail.com		2841 Rodney Street
kimkim.g.1980@gmail.com		2314 Pearl Avenue
tony.stanley.ca@gmail.com	Tony Stanley	2823 Percy Avenue
heleb.b.broom@gmail.com	Helen Broom	746 Happy Valley

Index Table

Column in Detail

Querying data with index

SORTING

Uploader Email
donald.t.lopresti@gmail.com
michele.r.herrmann@gmail.com
wanda.a.Allison@gmail.com
carol.faulkner.us@gmail.com
kimkim.g.1980@gmail.com
tony.stanley.ca@gmail.com
heleb.b.broom@gmail.com

Column in Detail

Querying data with index

Uploader Email
carol.faulkner.us@gmail.com
donald.t.lopresti@gmail.com
heleb.b.broom@gmail.com
kimkim.g.1980@gmail.com
michele.r.herrmann@gmail.com
⋮
tony.stanley.ca@gmail.com
wanda.a.Allison@gmail.com

Querying data with index

kimkim.g.1980@gmail.com

VS

tony.stanley.ca@gmail.com

Uploader Email
carol.faulkner.us@gmail.com
donald.t.lopresti@gmail.com
heleb.b.broom@gmail.com
kimkim.g.1980@gmail.com
michele.r.herrmann@gmail.com
⋮
tony.stanley.ca@gmail.com
wanda.a.Allison@gmail.com

Querying data with index

Uploader Email
carol.faulkner.us@gmail.com
donald.t.lopresti@gmail.com
heleb.b.broom@gmail.com
kimkim.g.1980@gmail.com
michele.r.herrmann@gmail.com
⋮
tony.stanley.ca@gmail.com
wanda.a.Allison@gmail.com

Uploader Email	Uploader Name	Uploader Address
donald.t.lopresti@gmail.com	Donald T. Lopresti	572 Lost Lake Crossing
michele.r.herrmann@gmail.com	Michele R. Herrmann	213 White Swan Garden
wanda.a.Allison@gmail.com	Wanda A. Allison	2017 Green Impasse
carol.faulkner.us@gmail.com	Carol Faulkner	2841 Rodney Street
kimkim.g.1980@gmail.com	Kimberly Greenlee	2314 Pearl Avenue
⋮	⋮	
tony.stanley.ca@gmail.com	Tony Stanley	2823 Percy Avenue
heleb.b.broom@gmail.com	Helen Broom	746 Happy Valley

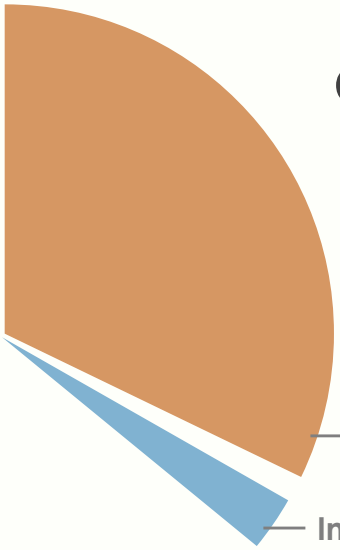
Column in Detail

Considerations of Using Index

Index – Use or Not?

Column in Detail

Considerations of Using Index



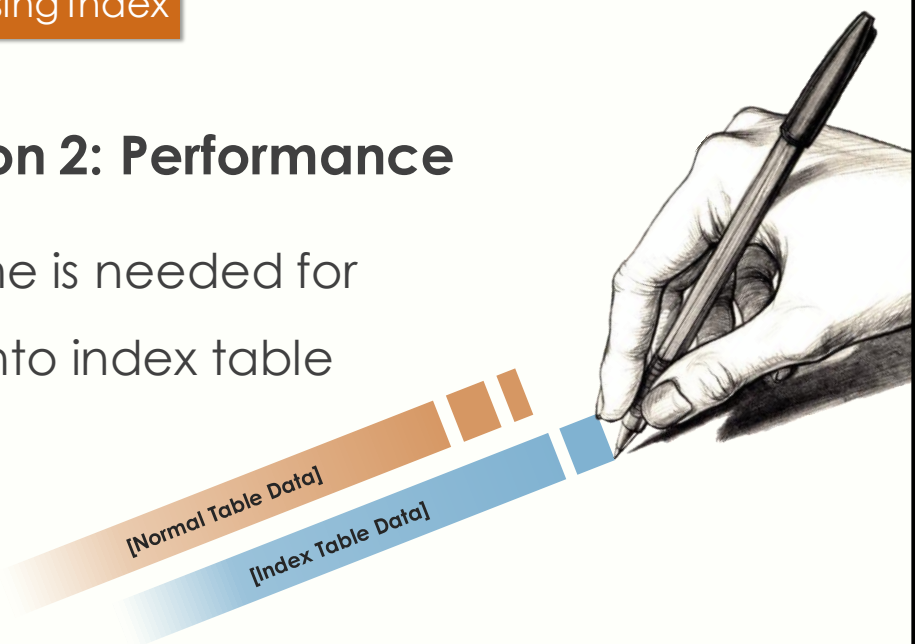
Consideration 1: Storage space

Index table consumes physical storage space

Considerations of Using Index

Consideration 2: Performance

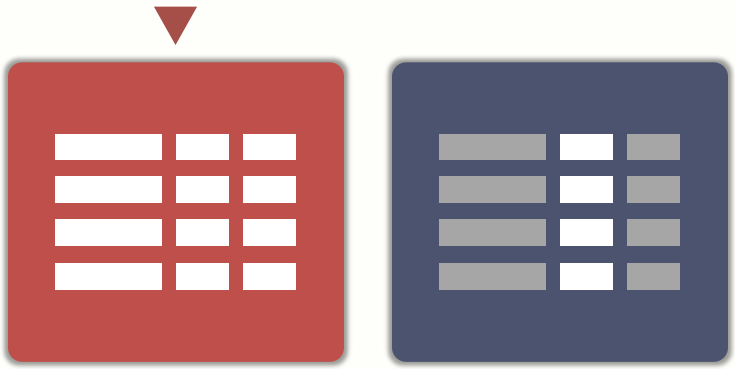
Additional time is needed for **writing data** into index table



Considerations of Using Index

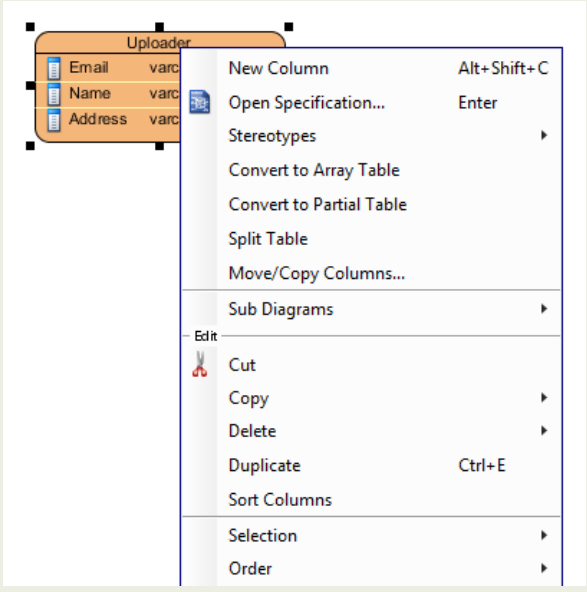
Index is used when we have an often need of **reading** data than **writing** data.

Defining index



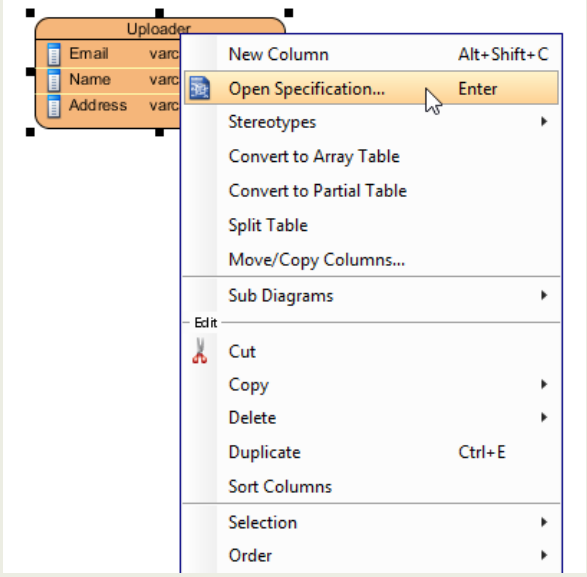
Defining index ■ Table space

1. Right click on an entity.



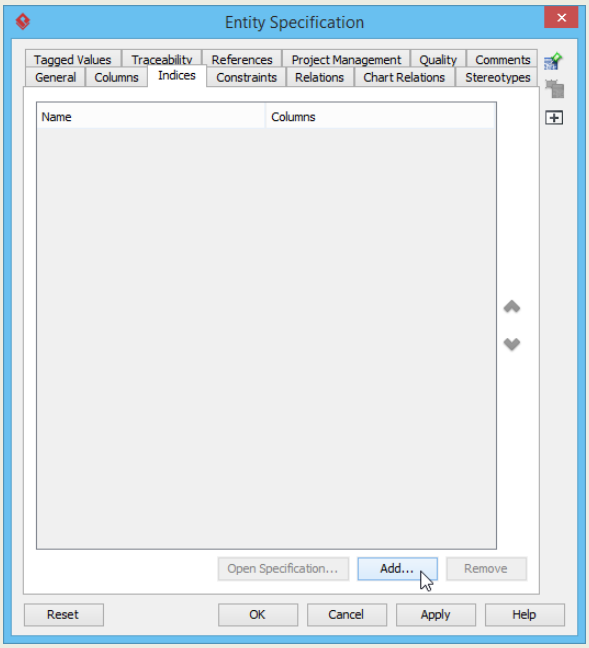
Defining index ■ Table space

1. Right click on an entity.
2. Select **Open Specification...** from the popup menu.



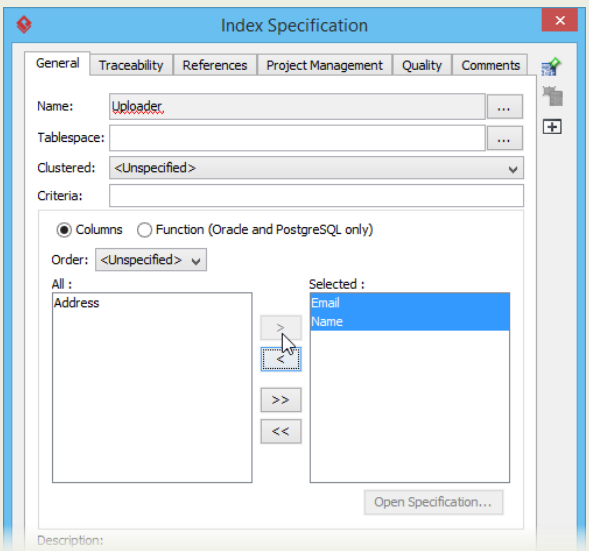
Defining index ■ Table space

1. Right click on an entity.
2. Select **Open Specification…** from the popup menu.
3. Click **Add…** in the **Indices** tab.



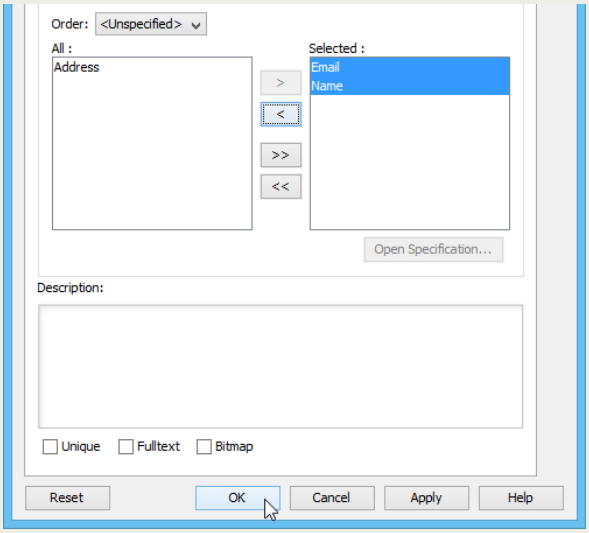
Defining index ■ Table space

1. Right click on an entity.
2. Select **Open Specification…** from the popup menu.
3. Click **Add…** in the **Indices** tab.
4. Select the columns to be included in index and click **>**.



Defining index ■ Table space

1. Right click on an entity.
2. Select **Open Specification...** from the popup menu.
3. Click **Add...** in the **Indices** tab.
4. Select the columns to be included in index and click **>**.
5. Click **OK** to confirm.



Column in Detail

Defining index ■ Column space

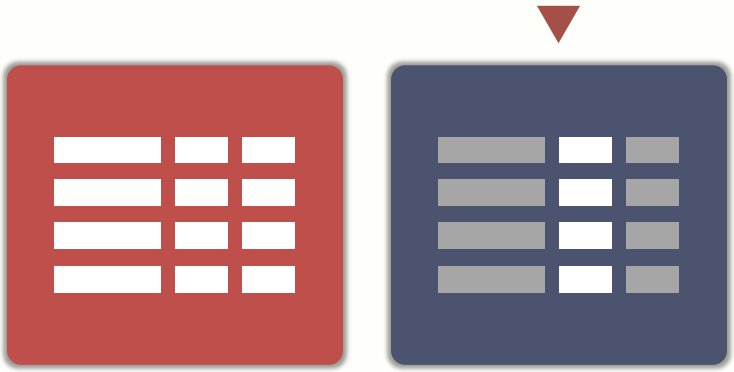


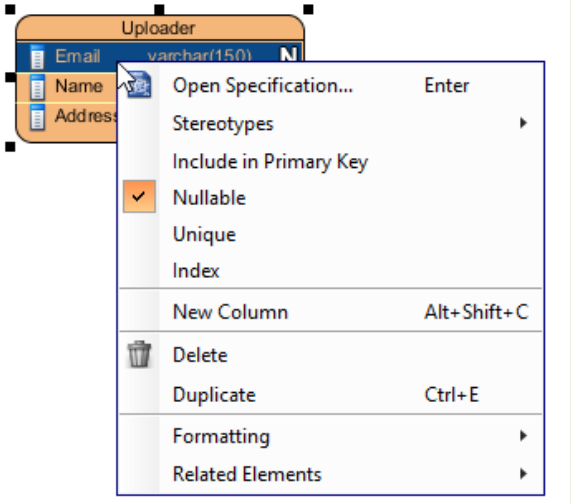
Table space
(Multiple columns in a table)

Column space
(Single column in a table)

Column in Detail

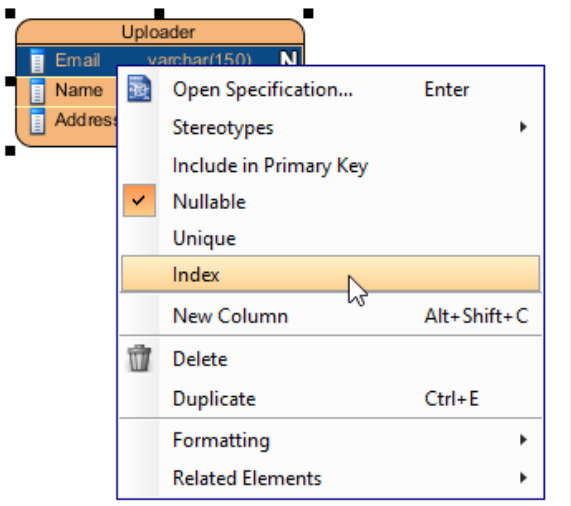
Defining index ■ Column space

1. Right click on a column.



Defining index ■ Column space

1. Right click on an entity.
2. Select **Index** from the popup menu.



Summary

- How to open column specification
- Various properties of an column
- The use of index