1. create a list of standard attributes and their possible values, uses in html.

Answer

1. **Content and Structure (High Usability):**

- `id` (**Values:** Unique identifier) - Uniquely identifies an element for styling and scripting.
- `class` (**Values:** Space-separated keywords) - Assigns a class name to an element for group styling and scripting.
- `lang` (**Values:** Language code (e.g., "en", "es")) - Specifies the language of the element's content.
- `title` (**Values:** Text) - Provides additional information displayed as a tooltip on hover.
- `style` (**Values:** CSS styles in curly braces) - Applies inline CSS styles directly to the element.

2. **Links and Navigation (High Usability):**

- `href` (**Values:** URL) - Defines the destination URL for hyperlinks (`<a>`) and base URL for `<base>`.
- `target` (**Values:** Window name, "_blank" (new window), "_self" (current window), etc.) - Specifies the target window or frame where a linked document should open.
- `rel` (**Values:** Link relationship type (e.g., "stylesheet", "alternate")) - Defines the relationship between the current document and the linked resource.
- `type` (**Values:** Media type (e.g., "text/css")) - Specifies the type of linked resource (e.g., stylesheet, script).

3. **Images and Multimedia (Medium Usability):**

- `src` (**Values:** URL) - Defines the path to the image file for `<img>` and the source of media for `<audio>` and `<video>`.
- `alt` (**Values:** Text description) - Provides alternative text for images, crucial for accessibility and SEO.
- `width & height` (**Values:** Positive integers with units (px, %, etc.)) - Sets the width and height of the displayed image or media.
- `poster` (**Values:** URL) - Specifies an image to be displayed before a video starts playing.

4. **Forms and User Input (Medium Usability):**

- `type` (**Values:** Varies based on input type (e.g., "text", "email", "checkbox")) - Defines the type of input field and how browsers handle the input.
- `name` (**Values:** Unique identifier) - Assigns a name to the input element for form submission and scripting.
- `value` (**Values:** Pre-filled text or selected value) - Sets the initial value displayed in the input field.
- `placeholder` (**Values:** Text) - Provides hint or instructional text displayed within the input field.
- `checked` (**Values:** Boolean (present or absent)) - Indicates a pre-selected checkbox or radio button.
- `required` (**Values:** Boolean (present or absent)) - Makes the input field mandatory (user must fill it before submitting the form).

5. **Tables (Low to Medium Usability - Depending on Usage):**

- `rowspan & colspan` (**Values:** Positive integers) - Defines how many rows/columns a table cell should span.
- `border` (**Values:** Integer specifying border width, or "collapse" for a collapsed border) - Sets the border style of the table.

- `caption` (**Values:** Text) - Adds a caption to the table.

6. **Other Useful Attributes:**

- `srcset` (**Values:** Comma-separated list of image sources with widths) - Provides options for responsive image selection based on screen size.
- `disabled` (**Values:** Boolean (present or absent)) - Disables an element (button, input field) making it uninteractable.
- `readonly` (**Values:** Boolean (present or absent)) - Makes an input field read-only (user can't edit the value).
- `placeholder` (**Values:** Text) - Provides hint or instructional text displayed within various elements (not just form inputs).
- `contenteditable` (**Values:** Boolean (true, false, "inherit")) - Allows users to edit the content of an element directly within the browser.

2. find out all possible properties, all possible values of a property and uses of CSS.

Text Properties:

- `color`: Defines text color (e.g., "red", "#ff0000", "rgb(255, 0, 0)").
- `font-family`: Specifies the font family (e.g., "Arial", "serif", "cursive").
- `font-size`: Sets the font size (e.g., "14px", "1em", relative units like %).
- `font-weight`: Controls font weight (e.g., "normal", "bold", numerical values).
- `text-align`: Aligns text within the element (e.g., "left", "center", "right").
- `text-decoration`: Applies decorative styles (e.g., "none", "underline", "overline").
- `line-height`: Sets the distance between lines of text (e.g., "1.5", units like px or em).
- `letter-spacing`: Adjusts spacing between characters (e.g., "normal", positive/negative values for px).

Box Model Properties:

- `margin`: Sets margins around the element (individual values for top, right, bottom, left, or shorthand for all). Units like px, em, or %.
- `padding`: Adds space within the element's border (individual values or shorthand for all). Units like px, em, or %.
- `border`: Defines the border style of the element (individual values for width, style, color, or shorthand for all). Units for width (px, em), color as with text color.
- `width & height`: Sets the width and height of the element (various units like px, em, %, or auto).

Background Properties:

- `background-color`: Sets the background color (same value options as text color).
- `background-image`: Defines a background image (URL referencing the image file).
- `background-repeat`: Controls how a background image repeats (e.g., "no-repeat", "repeat").
- `background-position`: Positions the background image (e.g., "center", "top left", using keywords or lengths).

Positioning Properties:

- `position`: Sets the positioning of the element (e.g., "static", "relative", "absolute", "fixed").
- `top`, `right`, `bottom`, `left`: Used with `position` to define the element's location relative to its container (units like px, em, %).

- `z-index`: Sets the stacking order of positioned elements (higher values appear on top).

List Properties:

- `list-style`: Sets list properties (e.g., "none", "disc", "circle", image URL for custom list style).
- `list-style-position`: Positions list item markers (e.g., "inside", "outside").

Flexbox Properties:

- `display`: Can be set to "flex" or "inline-flex" to enable flexbox layout.
- `flex-direction`: Defines the direction of flex items (e.g., "row", "column", "row-reverse").
- `justify-content`: Aligns flex items along the main axis (e.g., "flex-start", "center", "space-between").
- `align-items`: Aligns flex items along the cross axis (e.g., "flex-start", "center", "baseline").

Grid Properties:

- `display`: Can be set to "grid" to enable grid layout.
- `grid-template-columns`: Defines the layout of columns in the grid (e.g., "repeat(3, 1fr)", fractions, or minimum content).
- `grid-template-rows`: Defines the layout of rows in the grid (same options as columns).
- `grid-gap`: Sets the gap between grid items (units like px or em).