REGLAS BÁSICAS

- XQuery es “case-sensitive”
- Un valor de XQuery puede estar entre comillas simples o dobles
- Una variable XQueyr se define con $ seguido del nombre
- Los comentarios van delimitados por (: y :)

LOCALIZACIÓN DE NODOS

- Sigue la notación de XPath

EXPRESIONES FLOWR (para combinar y reestructurar nodos)

Definición: Una expresión FLOWR no está definida en términos de tablas, filas y columnas. Una expresión FLOWR asocia expresiones XPath a las variables de las cláusulas for y let, y utiliza dichas asociaciones para crear nuevos resultados. En XQuery, cuando usamos el término tupla, nos estamos refiriendo a cada uno de los valores que toma esa asociación

- **for**: asocia una o más variables a una secuencia de valores, creando una serie de tuplas en el que cada tupla esta vinculada a una de las variables
- **let**: asocia una variable al resultado entero de una expresión y devuelve una única tupla, pudiéndose añadir esta nueva asociación a las tuplas generadas por la cláusula for si esta existe
- **where**: filtra las tuplas producidas por las cláusulas let y for eliminando todos los valores que no cumplan las condiciones dadas. (opcional)
- **order by**: ordena las tuplas según el criterio dado (opcional)
- **return**: construye el resultado de la consulta para cada tupla dada, después de haber sido filtrada por la cláusula where y ordenada por la cláusula order by (obligatoria)

REGLAS GENERALES DE EXPRESIONES FLOWR

- for y let sirven para crear tuplas con las que trabajará el resto de las cláusulas de la consulta y pueden usarse tantas veces como se desee en una consulta, incluso dentro de otras cláusulas
- Sin embargo, solo puede declararse una única cláusula where (aunque puede tener condiciones anidadas por and, or, …), una única cláusula order by (aunque es posible especificar varios criterios de ordenación separándolos por comas) y una única cláusua return

OPERADORES Y FUNCIONES PRINCIPALES

- Matemáticos: +, -, *, div, idiv(división entera entre enteros), mod (resto)
- Comparación: =, !=, <, <=, >, >=, not()
- Secuencia: | (union), intersect, except
- Redondeo: floor(), ceiling(), round()
- Funciones de agrupación: count(), min(), max(), avg(), sum()
- Funciones de cadena: concat(), string-length(), starts-with(), ends-with(), substring(), upper-case(), lower-case(), string()
- Uso general:
    - distinct-values() devuelve una secuencia de valores atómicos distintos
    - emty() devuelve true si está vacío

- exists() devuelve true si no está vacío

CONSTRUCCIÓN DE NODOS

Para que la consulta devuelva un elemento o atributo, debemos incluirlo en la cláusula return de la siguiente manera:

- Etiquetas de apertura y cierre
- Y en medio y entre {} una lista de expresiones que devuelven el contenido del elemento (separadas por ',')

ITERACIONES

La cláusula for nos puede servir para repetir un número fijo de veces una acción

for $a in(1 to 5) → El return se ejecutaría 5 veces

EXPRESIONES CUATIFICADAS

Las expresiones cuatificadas nos sirven para testear si en una lista de elementos alguno (some) o todos (every) los elementos cumplen (satisfies) una condición, es decir, la salida de esta expresión es un booleano true/false

EXPRESIONES CONDICIONALES

Las expresiones condicionales en XQuery tienen la misma semántica que en los lenguajes de programación habituales. Tienen la estructura if-then-else. La cláusula else es obligatoria siempre, si no queremos que haga nada pondremos else()

FUNCIONES

Las funciones tienen la misma misión que en los lenguajes de programación comunes y tienen la estructura siguiente (recuerda que siempre tienen que terminar en ;):

declare function local:nombre_de_funcion($variable as xs:tipo_variable) as xs:tipo_devuelto

{

};