*--------------------------------------------------------------------------------------------------------*

*-- Ordering, Grouping Basic Functions Lecture Code*

*--------------------------------------------------------------------------------------------------------*

*--*

*-- The order of rows in the result (returned) is undefined unless you code an ORDER BY*

*--*

*-- The database manager does not guarantee the order of the rows in the result unless you have an ORDER BY*

*--*

*-- The same query with out an ORDER BY may return the same order of results 100 times in a row*

*-- the 101st time could be different*

*--*

*-- If the order of rows in the result matter - code an ORDER BY - DO NOT ASSUME THE ORDER IS ALWAYS THE SAME*

*--*

*-- null values are sorted as group at the beginning of result - if included*

*-- use IS NOT NULL to exclude null from the results*

*--*

*-- ORDER BY - Sequence of Rows in Result*

*--*

*-- ORDER BY -- Ascending Sequence (low-high)*

*-- ORDER BY ASC -- Ascending Sequence (low-high)*

*-- ORDER BY DESC -- Descending Sequence (high-low)*

*-- Show Populations of all countries in acscending order*

**select** **name**, population

**from** country

**order** **by** population **asc**

;

*-- Show Populations of all countries in descending order*

**select** population

**from** country

**order** **by** population **desc**

;

*-- Show Populations of all countries in acscending order only if population is greater than 1000000*

**select** **name**, population

**from** country

**where** population **>** 1000000

**order** **by** population **desc**

;

*-- Show the names of countries and continents in ascending order*

**select** **name**, continent

**from** country

**order** **by** continent **asc**

;

*-- Show population density for each country from highest to lowest*

**select** **name**

,population **/** surfacearea *-- derived column does not have a name*

**from** country

**order** **by** 2 *-- order by column number*

;

**select** **name**

,population **/** surfacearea **as** pop\_density *-- derived column can be assigned a name*

*-- ,population*

*-- ,surfacearea*

**from** country

**order** **by** pop\_density *-- order by using derived name*

;

*--------------------------------------------------------------------------------------------------------*

*-- Limiting the number of rows in the result*

*--*

*-- LIMIT n - Limit the number of rows in the result - always goes at the end of the SELECT*

*--*

*-- LIMIT is a postgreSQL word - all database managers have a way to LIMIT results - might called something else*

*--*

*-- Show the name and average life expectancy of the countries with the 10 highest life expectancies.*

*-- distinct shows only unique values*

*--*

**select** **distinct** **name**, lifeexpectancy *-- distinct is applied to the combination of all columns in the select*

**from** country

**where** lifeexpectancy **is** **not** **null**

**order** **by** lifeexpectancy **desc**

**limit** 10 *-- only show the first 10 rows from the result*

;

*--*

*-- List all the unique continent names*

*--*

**select** **distinct** continent *-- distinct is applied to the combination of all columns in the select*

**from** country

;

*--------------------------------------------------------------------------------------------------------*

*-- Concatenating values*

*--*

*-- the concat operator (||) may be used to concatenate character (string) values in a result*

*--*

*-- Show the name & state in the format: "city-name, state"*

*-- of all cities in California, Oregon, or Washington.*

*-- sorted by state then city*

**select** **name** **||** ', ' **||** district **as** city\_name *-- Column in result*

**from** city

**where** district **=** 'California'

**or** district **=** 'Oregon'

**or** district **=** 'Washington'

**order** **by** district, **name** *-- order of rows in the result*

;

**select** **name** **||** ', ' **||** district **as** city\_name *-- Column in result*

**from** city

**where** district **in**('California', 'Oregon','Washington')

**order** **by** district, **name** *-- order of rows in the result*

;

*--------------------------------------------------------------------------------------------------------*

*-- Aggregate functions - produce one row in result for each group specified*

*--*

*-- The group used by the aggregate functions is determined by the GROUP BY clause*

*-- if no GROUP BY clause is specified, the group is the set of rows in the result*

*--*

*-- AVG(column-expression) - arithmentic average for group of non-NULL values in expression*

*-- SUM(column-expression) - arithmentic sum for group of a non-NULL values in expression*

*-- MIN(column-expression) - lowest value for group of non-NULL values in expression*

*-- MAX(column-expression) - highest value for group of non-NULL values in expression*

*-- COUNT(\*) - number of rows in the group*

*-- COUNT(column-expression) - number of rows for the group of non-NULL values in expression*

*--*

*-- AVG, MIN, MAX, SUM - work on columns - AVG(column) Sum (column)*

*--*

*-- COUNT - works on rows - how many rows?*

*--*

*-- AVG(), SUM() may only be used with numeric data types*

*-- MIN(), MAX() may be used with numeric, character, date, time datatypes*

*--*

*-- COUNT() is applied to rows (not columns)*

*--*

*-- null values are ignored by all aggregate functions*

*--*

*--*

*-- Show average life expectancy in the world*

**select** **avg**(lifeexpectancy) *-- avg of non-null lifeexpecancy of the entire table*

**from** country

; *-- no group by specified so on value for entire table is generated*

*-- Show average life expectancy of each continent*

**select** continent, **avg**(lifeexpectancy) *-- avg of non-null lifeexpecancy*

**from** country

**where** lifeexpectancy **is** **not** **null** *-- don't include any rows if lifeexpectancy is unknown*

**group** **by** continent *-- Specify the group for the aggregate function - one value for each continent*

;

*-- Show the total population in Ohio*

**select** **sum**(population) **as** Ohio\_Population

**from** city

**where** district **=** 'Ohio'

;

*-- Show the total population in Ohio*

*--*

*-- When using a Group By the only columns allowed in the SELECT are columns*

*-- named in the group by or in in the functio.*

*--*

*-- column in the SELECT tells SQL you want one row in the result for every row that matches the where clause*

*-- GROUP BY tells SQL you want one row in the result for each disctinct value for column in the function*

*--*

*-- conflicting requests when column not in group by is coded with aggregate function*

*--*

*--*

*-- This SQL causes an error - combines a column and aggregate function on the same select without a group by*

*--*

*--select name, sum(population) as Ohio\_Population*

*-- from city*

*--where district = 'Ohio'*

*--;*

*-- Show the total population in Ohio*

**select** **sum**(population) **as** Ohio\_Population

,**avg**(population) **as** avg\_population

,**count**(**\***) **as** Num\_Ohio\_Cities

,**min**(population) **as** Smallest\_Ohio\_Pop

,**max**(population) **as** Largest\_Ohio\_Pop

,**min**(population) **+** **max**(population) **as** Billy\_Pop

**from** city

**where** district **=** 'Ohio'

;

*-- Show the surface area of the smallest country in the world*

**select** **name**, surfacearea **as** Smallest\_Area

**from** country

**order** **by** Smallest\_Area **asc** *-- combination of order by*

**limit** 1 *-- and limit 1 can be used in place of min/max*

;

*-- Show The 10 largest countries (by surface area) in the world*

*-- Paul's choice*

**select** **name**, **max**(surfacearea) **as** largest\_area

**from** country

**group** **by** **name**

**order** **by** largest\_area **desc**

**limit** 10;

*-- Matthias' choice*

**select** **name**, surfacearea

**from** country

**order** **by** surfacearea **desc**

**limit** 10;

*--Nate's choice*

**select** **name**, surfacearea

**from** country

**order** **by** surfacearea **desc**

**limit** 10;

*-- Show the number of countries who declared independence in 1991*

**select** **count**(**\***) *-- \* means all the rows*

**from** country

**where** indepyear **=** 1991

;

**select** **count**(**name**) *-- any column name can be included in count*

**from** country

**where** indepyear **=** 1991

;

*--------------------------------------------------------------------------------------------------------*

*-- GROUP BY - Specify the group to which the aggregate functions apply*

*--*

*-- GROUP BY column-expression*

*--*

*-- When using a GROUP BY the SELECT is limited ot aggreggate functions or columns in the GROUP BY*

*--*

*--*

*-- Show the number of countries where each language is spoken, order show them from most countries to least*

**select** **count**(**\***), **language**

**from** countrylanguage

**group** **by** **Language**

**order** **by** **count**(**\***) **desc**

;

*-- Show the average life expectancy of each continent ordered from highest to lowest*

*-- Exclude Antarctica from consideration for average life expectancy*

*-- What is the sum of the population of cities in each state in the USA ordered by state name*

*-- What is the average population of cities in each state in the USA ordered by state name*

*--------------------------------------------------------------------------------------------------------*

*-- SUBQUERIES - Using the result from one query (inner query) in another query (outer query)*

*--*

*-- Frequently used in a WHERE clause with an IN predicate:*

*--*

*-- WHERE column-name IN (SELECT column-name FROM some-table WHERE some-predicate)*

*--*

*-- Any WHERE predicate may be used to connect the subquery in a WHERE clause, but you must*

*-- be sure a single value is returned from the subquery.*

*--*

*-- Subqueries may also be used in a SELECT as a column-specification or a FROM as a table*

*-- (These are advanced concepts we will discuss later, if there is time)*

*--*

*-- Show the cities under the same given government leader*

*-- Show countries with the same independece year*

*-- Show the cities cities whose country has not yet declared independence yet*

*--------------------------------------------------------------------------------------------------------*

*--*

*-- Additional samples*

*--*

*-- You may alias column and table names to provide more descriptive names*

*--*

**SELECT** **name** **AS** CityName

**FROM** city **AS** cities

*-- Ordering allows columns to be displayed in ascending order, or descending order (Look at Arlington)*

**SELECT** **name**

, population

**FROM** city

**WHERE** countryCode**=**'USA'

**ORDER** **BY** **name** **ASC**, population **DESC**

;

*-- Limiting results allows rows to be returned in 'limited' clusters where LIMIT says how many,*

*-- and an optional OFFSET specifies number of rows to skip*

**SELECT** **name**

, population

**FROM** city

**LIMIT** 10 **OFFSET** 10

;