**SQL Notes**

BASIC STATEMENTS

* SELECT what FROM where - selects particular data from database (table)

‘\*’ can be used to select everything (i.e., all columns)

* SELECT DISTINCT what FROM where - what are the unique values from the column
* SELECT COUNT(what) FROM where – count the number of rows
* SELECT COUNT(DISTINCT(what)) FROM where
* SELECT what FROM where

WHERE ‘logical operators & conditional statements’

* … ORDER BY column ASC(default)/DESC
* …LIMIT – the last column past in the query
* …(NOT) BETWEEN (inclusive for between, not inclusive for the NOT between)
* …IN(option1,option2…)
* …LIKE ‘A%’ % stands for any number (even zero) of characters (case sensitive)

…LIKE ‘A\_’ % stands for a single character (case sensitive)

…ILIKE ‘A%’ case insensitive

…LIKE ‘A\_’ % stands for a single character (case insensitive)

* …GROUP BY column(s) – groups the results by the column, the column in questions has to be first selected in the SELECT statement, filtering has to happen before the group by statement
* …DATE (date column) – selects date only from a time stamp column
* …HAVING – if we want to filter based on the aggregate result (i.e, after group by)

JOINS

* …AS create an alias for output (assigned at the very end!)
* SELECT \* FROM table 1

INNER JOIN table2 ON table1.column\_name = table2.column\_name – select common values from both tables (column name has to be provided)

* \* can be replaced with the actual column names that are to be selected (i.e., if we do not want the column to be repeated from both table 1 and table 2)
* SELECT \* FROM table 1

OUTER JOIN table2 ON table1.column\_name = table2.column\_name – grabs everything (places null where values are missing)

* … WHERE table1.column IS null OR table2.column IS null – the opposite of inner join
* …LEFT OUTER JOIN – grabs only the things that are exclusive to the first table (fills in null for missing values in table 2) or common to both but not anything that is exclusive to the second table
* …WHERE table2 IS null – shows values that are only in table 1 but not in table 2
* …RIGHT OUTER JOIN- same as left outer join but with reverse order of the tables
* SELECT…UNION…SELECT – union of two select statements along rows

ADVANCED SQL COMMANDS

* EXTRACT()
* AGE()
* TO\_CHAR()
* Mathematical operators and functions
* String functions and operators
* SubQuery – query within another query (i.e., series of two SELECT, subselect is in paratheses)
* EXIST- checks if exists in the output table
* SELF-JOIN- same as join but necessary to use aliases
* CURRENT\_TIMESTAMP- provides current time stamp

CREATING DATABASES AND TABLES

* Primary and Foreign keys (can be found in constraints 🡪 dependencies)
* Constrains (column or whole table) – NOT NULL, UNIQUE, PRIMARY KEY, FOREIGN KEY, CHECK, EXCLUSION
* CREATE TABLE table\_name(column\_name TYPE column\_constraint, …, table\_constraint table\_constraint) INHERITS existing\_table\_name;
* TYPE- SERIAL (very useful for primary key)
* e.g. CREATE TABLE account(user\_id SERIAL PRIMARY KEY, username VARCHAR(50) UNIQUE NOT NULL, password VARCHAR(50) NOT NULL, email VARCHAR(250) UNIQUE NOT NULL, created\_on TIMESTAMP NOT NULL, last\_login TIMESTAMP)
* for linking table:

CREATE TABLE table\_name(column\_name REFERENCES reference\_table1\_name(column), column\_name REFERENCES reference\_table2\_name(column))

* Add new rows to an existing table:

INSERT INTO table(column1,column2,…) VALUES (value1,value2,…), (value1,value2),…

* Insert from another table:

INSERT INTO table(column1,column2,…) SELECT column1,column2,… FROM another\_table WHERE condition

* UPDATE table SET column1=value1,… [WHERE condition] [ ]=optional
* (UPDATE join)- i.e., from another table: UPDATE table SET column1=value1 FROM tableB WHERE tableA.id = tableB.id
* RETURNING columns – if added at the end will show all the listed columns after update
* DELETE FROM table – delete all rows
* DELETE FROM tableA USING tableB WHERE tableA.id = tableB.id
* Allows changes to an existing table:

ALTER TABLE table\_name

ALTER COLUMN column\_name

e.g., RENAME TO new\_name, RENAME COLUMN old TO new, check documentation for all the options!

* DROP – removes a complete column from a table

CASCADE clause also remove all the dependencies on the removed column

e.g., ALTER TABLE table\_name DROP COLUMN [IF EXISTS] column\_name [CASCADE]

* CHECK- allows for checking for certain constraints

e.g., CREATE TABLE table\_name(column\_name SERIAL PRIMARY KEY, age SMALLINT CHECK (age>21), parent\_age SMALLINT CHECK (parent\_age>age));

CONDITIONAL EXPRESSIONS AND PROCEDURES

* CASE (similar to if/else)
* CASE general syntax:

SELECT column,

CASE

WHEN condition1 THEN result1

WHEN condition2 THEN result2

ELSE some\_other\_result

END

FROM table;

Conditions can be complex conditions (e.g., customer\_id > 100)

* CASE expression :

SELECT column,

CASE

WHEN x THEN result1

WHEN y THEN result2

ELSE some\_other\_result

END

X and y are only simple equalities (e.g., when 2 then ‘winner’)

* Functions can be called on the CASE statements
* COALESCE(arg\_1,arg\_2,arg\_3): e.g., SELECT item,(price-COALESCE(discount,0)) AS final FROM table
* CAST- allows to convert from one data type into another (within reasonable limits): SELECT CAST(‘5’ AS INTEGER)
* NULLIF returns a null if both arguments are equal (e.g., to avoid dividing by zero, or check that something is equal to zero)
* CREATE [OR REPLACE] VIEW name\_of\_the\_virtualtable AS … - view is a database object that is of a stored query (i.e., virtual table)
* DROP VIEW, ALTER VIEW…
* Import and Export- first a table needs to be created and then will be populated from the imported file- right click on table Import/Export and fill in the details