Introduction to SQL

In the modern world, the data is stored in relational databases. Now to fetch or store the specific data from the databases, we need to develop a methodology or set of rules. This set of rules is collectively called Structured Query Language or SQL. In this section, we will discuss major queries and what they do.

How does it work in real life?
Databases live on a server, which manages them. Users interact with the server through a client program. It allows multiple users access the same database simultaneously

SQL mostly writes standard queries like the *dplyr* library does in R. For example, consider the following code:

flights %>% filter(carrier == "UA") %>% select(origin, dest)

What this does is filter and select some observations based on the inputs given. SQL does select and filter work. This is a very simple language but is very fast, therefore helps us fetch the data of large size.

SELECT query

SELECT is the first word of a query, then modifiers say which fields/columns to use, from which table(s), and what conditions records/rows must meet. We must add a final semi-colon in the query. Some example of a simple SELECT query are illustrated below:

1. Task: Selecting two columns ‘origin’ and ‘dest’ from table `flights`

   SELECT origin, dest FROM flights;

2. Task: Selecting All columns from table `flights`

   *SELECT * FROM flights;*

3. Task: Selecting as above, but by ascending value of `arr_delay`

   *SELECT * FROM flights ORDER BY arr_delay;*

4. Task: Selecting worst arrival delayed 10 flights

   *SELECT * FROM flights ORDER BY arr_delay DESC LIMIT 10;*

5. Task: Picking out rows meeting a condition
   SELECT origin, dest
   FROM flights
   WHERE arr_delay > 100 AND dep_delay > 0;

6. Task: Calculations on value-grouped subsets, like in group_by summarise in dplyr (or aggregate or d*ply in base R)
   SELECT origin, AVG(dep_delay)
   FROM flights
   GROUP BY origin;

<u>JOIN query</u>

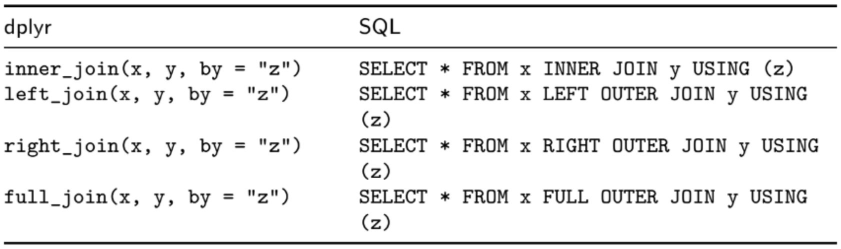
Join query joins the desired table. There are four types of joins: inner, outer/full, left, right.

1. Task-1: Join if the names are the same in the two tables, join them using inner join:
   SELECT year, month, day, carrier, flight
   FROM flights INNER JOIN airlines
   USING(carrier);

2. Task-2: Join if the names are different in the two tables
   SELECT year, month, day, carrier, flight, origin, dest
   FROM flights
   INNER JOIN airports
   ON flights.dest == airports.faa
   WHERE flights.origin == "JFK"
   AND airports.tz <= -8;

<u>Relating SQL to R:</u>

There are many such queries. For a programmer working in R or python, SQL should be very easy. Here we provide a table to relate SQL queries with R.



dplyr	SQL
inner_join(x, y, by = "z")	SELECT * FROM x INNER JOIN y USING (z)
left_join(x, y, by = "z")	SELECT * FROM x LEFT OUTER JOIN y USING (z)
right_join(x, y, by = "z")	SELECT * FROM x RIGHT OUTER JOIN y USING (z)
full_join(x, y, by = "z")	SELECT * FROM x FULL OUTER JOIN y USING (z)