Exploratory Data Analysis with R

Basic SQL Queries - Part I

Xuemao Zhang
East Stroudsburg University

November 28, 2022

Outline

- Basic SQL Commands
  - CREATE DATABASE - creates a new database.
  - CREATE TABLE - creates a new table.
  - SELECT - extracts data from a database.
  - ALTER and ADD - add a column in an existing table.
  - UPDATE - updates data in a database.
  - DELETE - deletes data from a database.
  - INSERT INTO - inserts new data into a database.
  - CREATE INDEX - create indexes in tables

SQL Key Words

- SQL Tutorial https://www.w3schools.com/sql/sql_syntax.asp

Some of The Most Important SQL **Key Words**:

- `SELECT` - extracts data from a database
- `UPDATE` - updates data in a database
- `DELETE` - deletes data from a database
- `INSERT INTO` - inserts new data into a database
- `CREATE DATABASE` - creates a new database
- `ALTER DATABASE` - modifies a database
- `CREATE TABLE` - creates a new table
- `ALTER TABLE` - modifies a table
- `DROP TABLE` - deletes a table
- `CREATE INDEX` - creates an index (search key)
- `DROP INDEX` - deletes an index

Basic SQL Commands - SELECT

SELECT	desired attributes
FROM	one or more tables
WHERE	conditions on rows of the tables are satisfied

- Retrieving an entire table

```
SELECT * FROM surveys;
```

- Return the first 100 rows only using key word `LIMIT`

```
SELECT * FROM surveys LIMIT 100;
```

Basic SQL Commands - SELECT

- Select columns

```
SELECT sex, hindfoot_length FROM surveys;
```

- Find unique sexs using `DISTINCT`

```
SELECT DISTINCT sex FROM surveys;
```

Basic SQL Commands - SELECT

- The `ORDER BY` keyword is used to sort the result-set in ascending or descending order.

SELECT	*
FROM	table
ORDER BY	columns ASC|DESC;

- Oder the data by variable `weight`

```
SELECT * FROM surveys
ORDER BY weight ASC;
```

Basic SQL Commands - SELECT

- Select columns with `weight <= 5`

```
SELECT species_id, sex, hindfoot_length, weight
FROM surveys
WHERE weight <= 5;
```

- A boolean expression in the **WHERE** clause may contain the following operators or any combination:
  - `AND`
  - `NOT`
  - `OR`

Basic SQL Commands - SELECT

- Select columns with `weight <= 5` and `sex==M`

```
SELECT species_id, sex, hindfoot_length, weight
FROM surveys
WHERE weight <= 5 AND sex='M';
```

- Select columns with `weight` between 5 and 6

```
SELECT species_id, sex, hindfoot_length
FROM surveys
WHERE weight >= 5 AND weight <= 6;
```

Basic SQL Commands - SELECT

- Select columns with sex is not Female

```
SELECT species_id, sex, hindfoot_length, weight
FROM surveys
WHERE NOT sex='F';
```

- IS NULL Syntax

```
SELECT species_id, sex, hindfoot_length, weight
FROM surveys
WHERE weight is NULL;
```

- IS NOT NULL Syntax

```
SELECT species_id, sex, hindfoot_length, weight
FROM surveys
WHERE weight is NOT NULL;
```

Basic SQL Commands - SELECT

- Select columns with `species_id` is DM or DO

```
SELECT species_id, sex, hindfoot_length, weight
FROM surveys
WHERE species_id='DM' OR species_id='DO';
```

- Select columns with `species_id` is DM or DO and `sex=F`

```
SELECT species_id, sex, hindfoot_length, weight
FROM surveys
WHERE (species_id='DM' OR species_id='DO') AND sex='F';
```

Basic SQL Commands - SELECT

- The `SELECT INTO` statement copies data from one table into a new table.

SELECT	*
INTO	newtable [IN externaldb]
FROM	oldtable
WHERE	conditions

- For example, we save the data with NULL in the columns `hindfoot_length` and `weight` removed

```
SELECT *
INTO surveys_weight
FROM surveys
WHERE weight is NOT NULL AND hindfoot_length is NOT NULL;

SELECT * FROM surveys_weight;
```

Basic SQL Commands - SELECT

- The `GROUP BY` statement groups rows that have the same values into summary rows.

SELECT	column1 aggregate(column2)
FROM	table name
WHERE	condition
GROUP BY	column1
ORDER BY	column1;

```
SELECT sex, avg(hindfoot_length), avg(weight)
FROM surveys
GROUP BY sex;
```

- Aggregate Functions: https://www.postgresql.org/docs/9.5/functions-aggregate.html

Basic SQL Commands - SELECT

- Some other arithmetic summaries: Min, Max, Count and Sum

```
SELECT  min(weight) as min_weight, max(weight) as max_weight,
count(weight) as n_weight, sum(weight) as sum_weight
FROM surveys;
```

- We can use statistical summaries as well. For example,

```
SELECT  corr(hindfoot_length, weight)
FROM surveys;
```

Basic SQL Commands - SELECT

- The `DROP TABLE` statement is used to drop an existing table in a database.

```
DROP TABLE surveys_weight;
```

- The `CREATE DATABASE` statement is used to create a new SQL database.
  CREATE DATABASE databasename;
- The `DROP DATABASE` statement is used to drop an existing SQL database.
  DROP DATABASE databasename;

- Let's create a database `Database2` and then delete it

```
CREATE DATABASE Database2;
DROP DATABASE IF EXISTS Database2;
```

Basic SQL Commands - ALTER and ADD

- The `ALTER TABLE` statement is used to add, delete, or modify columns in an existing table.

ALTER TABLE	table name
ADD	column name datatype;

- For example, we add another measure `tail_length`

```
ALTER TABLE surveys
ADD tail_length integer;
```

Basic SQL Commands - UPDATE

- The `UPDATE` statement is used to modify the existing records in a table.

UPDATE	table name
SET	column1 = value1, column2 = value2, ...
WHERE	condition;

Note: The `WHERE` clause specifies which record(s) that should be updated. If you omit the `WHERE` clause, all records in the table will be updated! You cannot undo an update unless you ran it inside a transaction(https://www.postgresql.org/docs/8.3/tutorial-transactions.html).

- Let's update the first record

```
UPDATE surveys
SET sex = 'F', weight= 30
WHERE record_id = 1;

select * from surveys
where record_id = 1;
```

Basic SQL Commands - DELETE

- The `DELETE` statement is used to delete existing records in a table.

DELETE FROM	table name
WHERE	condition;

Note: The `WHERE` clause specifies which record(s) that should be deleted. If you omit the `WHERE` clause, all records in the table will be deleted! Again, you cannot undo a delete.

- Let's delete the first record

```
DELETE FROM surveys
WHERE record_id = 1;

select * from surveys
where record_id = 1; /*an empty row will be returned*/
```

Basic SQL Commands - INSERT INTO

- The `INSERT INTO` statement is used to insert new records/rows in a table.

1. Specify both the column names and the values to be inserted:
   INSERT INTO table name (column1, column2, column3, ...)
   VALUES (value1, value2, value3, ...);
2. If you are adding values for all the columns of the table, you do not need to specify the column names in the SQL query. However, make sure the order of the values is in the same order as the columns in the table.
   INSERT INTO table name
   VALUES (value1, value2, value3, ...);

- Let's insert one more row
  - Be careful, the Primary Key cannot be NULL

```
INSERT INTO surveys (record_id, sex, hindfoot_length, weight)
VALUES (1, 'F', 34, 40);

select * from surveys
where record_id = 1;
```

Basic SQL Commands - CREATE INDEX

- Indexes are used to retrieve data from the database more quickly than otherwise.
  - The users cannot see the indexes, they are just used to speed up searches/queries.
  - https://www.postgresql.org/docs/15/sql-createindex.html
  - https://www.postgresqltutorial.com/postgresql-indexes/postgresql-list-indexes/
- **Note**: Updating a table with indexes takes more time than updating a table without (because the indexes also need an update). So, only create indexes on columns that will be frequently searched against.

1. Creates an index on a table. Duplicate values are allowed:

CREATE INDEX	index name
ON	table name (column1, column2, ...);

2. Creates a unique index on a table. Duplicate values are not allowed:

CREATE UNIQUE INDEX	index name
ON	table name (column1, column2, ...);

Basic SQL Commands - CREATE INDEX

- The SQL statement below creates an index named `idx_year` on the `year` column in the `surveys` table:

```
CREATE INDEX idx_year
ON surveys (year);
```

- If you want to create an index on a combination of columns, you can list the column names within the parentheses, separated by commas:

```
CREATE INDEX idx_time
ON surveys (month, day, year);
```

Basic SQL Commands - CREATE INDEX

- An index can be dropped using the `DROP` command.

DROP INDEX index name;

```
DROP INDEX idx_year;
DROP INDEX idx_time;
```

Basic SQL Commands - CREATE INDEX

Testing Index performance:

- To test if indexes will begin to decrease query times, you can run a set of queries on your database, record the time it takes those queries to finish, and then begin creating indexes and rerunning your tests.
  - To do this, try using the EXPLAIN ANALYZE clause in PostgreSQL

```
EXPLAIN ANALYZE SELECT *
FROM surveys
WHERE month=7 AND day=16 AND year=1977;
```

License