# Exploratory data analysis using SQL

*Noushin Nabavi*

*2019-05-01*

## What is in a database?

### explore data table:

- Select the count of the number of rows

```
SELECT count(*)
  FROM tablename;
```

### counting missing data:

- Select the count of ticker,
- subtract from the total number of rows,
- and alias as missing

```
SELECT count(*) - count(ticker) AS missing
  FROM fortune500;
```

- Select the count of profits_change,
- subtract from total number of rows, and alias as missing

```
SELECT count(*) - count(profits_change) AS missing
  FROM fortune500;
```

- Select the count of industry,
- subtract from total number of rows, and alias as missing

```
SELECT count(*) - count(industry) AS missing
  FROM fortune500;
```

### joining tables:

```
SELECT company.name
-- Table(s) to select from
  FROM company
       INNER JOIN fortune500
       ON company.ticker=fortune500.ticker;
```

### The keys to the database (e.g. foreign vs. primary keys)

- Read an entity relationship diagram

```
-- Count the number of tags with each type
SELECT type, count(*) AS count
  FROM tag_type
 -- To get the count for each type, what do you need to do?
 GROUP BY type
```

```
 -- Order the results with the most common
 -- tag types listed first
 ORDER BY count DESC;
```

- or:

```
-- Select the 3 columns desired
SELECT name, tag_type.tag, tag_type.type
  FROM company
       -- Join the tag_company and company tables
       INNER JOIN tag_company
       ON company.id = tag_company.company_id
       -- Join the tag_type and company tables
       INNER JOIN tag_type
       ON tag_company.tag = tag_type.tag
  -- Filter to most common type
  WHERE type='cloud';
```

- coalesce function (to combine columns)

```
-- Use coalesce
SELECT coalesce(industry, sector, 'Unknown') AS industry2,
       -- Don't forget to count!
       count(*)
  FROM fortune500
-- Group by what? (What are you counting by?)
 GROUP BY industry2
-- Order results to see most common first
 ORDER BY count DESC
-- Limit results to get just the one value you want
 LIMIT 1;
```

- Coalesce with a self-join:

```
SELECT company_original.name, title, rank
  -- Start with original company information
  FROM company AS company_original
       -- Join to another copy of company with parent
       -- company information
       LEFT JOIN company AS company_parent
       ON company_original.parent_id = company_parent.id
       -- Join to fortune500, only keep rows that match
       INNER JOIN fortune500
       -- Use parent ticker if there is one,
       -- otherwise original ticker
       ON coalesce(company_parent.ticker,
                   company_original.ticker) =
            fortune500.ticker
 -- For clarity, order by rank
 ORDER BY rank;
```

## Column types and constraints

- Effects of casting
- SELECT CAST(value AS new_type);

```
-- Select the original value
SELECT profits_change,
       -- Cast profits_change
       CAST(profits_change AS integer) AS profits_change_int
  FROM fortune500;
```

```
-- Divide 10 by 3
SELECT 10/3,
       -- Divide 10 cast as numeric by 3
       10::numeric/3;
```

- SELECT value::new_type

```
SELECT '3.2'::numeric,
       '-123'::numeric,
       '1e3'::numeric,
       '1e-3'::numeric,
       '02314'::numeric,
       '0002'::numeric;
```

- Summarize the distribution of numeric values

```
-- Select the count of each value of revenues_change
SELECT revenues_change, count(*)
  FROM fortune500
 GROUP BY revenues_change
 -- order by the values of revenues_change
 ORDER BY revenues_change;
```

```
-- Count rows
SELECT count(*)
  FROM fortune500
 -- Where...
 WHERE revenues_change > 0;
```

# Numeric data types and summary functions

## Division

```
-- Select average revenue per employee by sector
SELECT sector,
       avg(revenues/employees::numeric) AS avg_rev_employee
  FROM fortune500
 GROUP BY sector
 -- Use the alias to order the results
 ORDER BY avg_rev_employee;
```

- explore by division

```
-- Divide unanswered_count by question_count
SELECT unanswered_count/question_count::numeric AS computed_pct,
       -- What are you comparing the above quantity to?
       unanswered_pct
  FROM stackoverflow
 -- eliminate rows where question_count is not 0
 WHERE question_count != 0
```

```
LIMIT 10;
```