POIR 613: Computational Social Science

Pablo Barberá

University of Southern California
pablobarbera.com

Course website:
pablobarbera.com/POIR613/

Plan for today

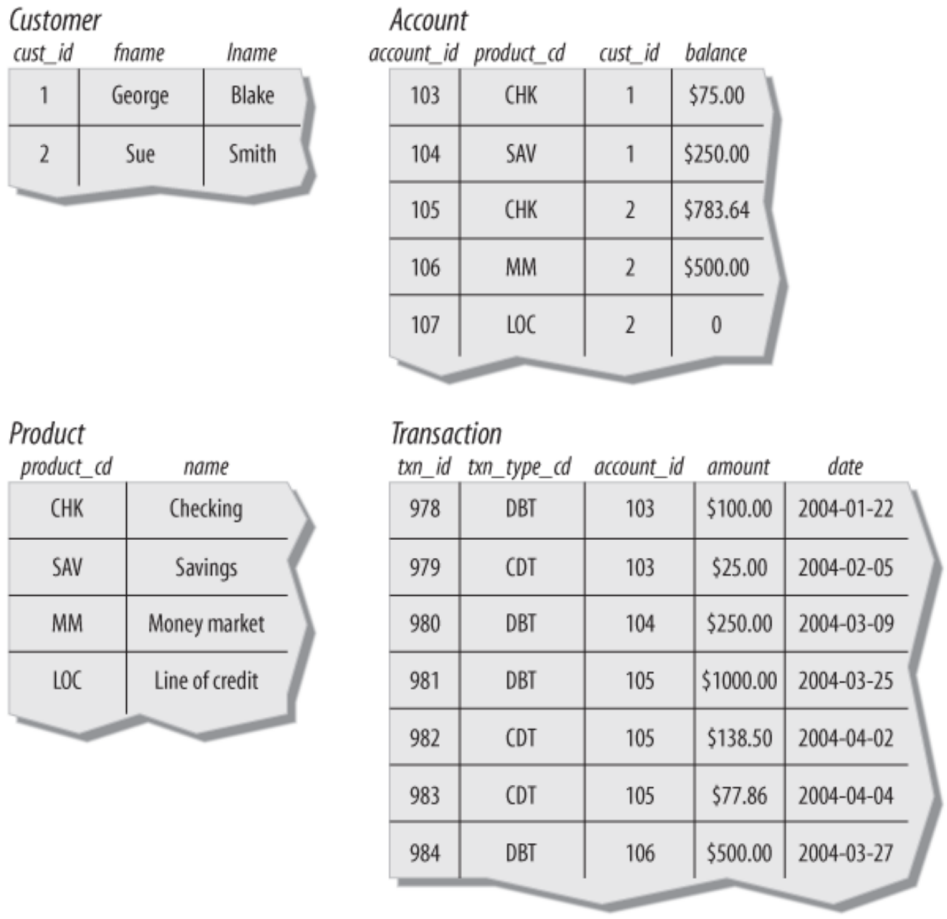
SQL

- Database systems
- Why SQL?
- Components of a SQL query
- Google BigQuery
- Guided coding sessions:
  1. Introduction
  2. JOINs and aggregations
  3. Querying large-scale datasets

Introduction to SQL

Databases

- Database systems: computerized mechanisms to store and retrieve data.
- Relational databases: type of database where data is represented as tables linked based on common keys (to avoid redundancy).
- Tables: database objects that hold the data. A database is thus a collection of tables.



Customer

cust_id	fname	lname
1	George	Blake
2	Sue	Smith

Account

account_id	product_cd	cust_id	balance
103	CHK	1	$75.00
104	SAV	1	$250.00
105	CHK	2	$783.64
106	MM	2	$500.00
107	LOC	2	0

Product

product_cd	name
CHK	Checking
SAV	Savings
MM	Money market
LOC	Line of credit

Transaction

txn_id	txn_type_cd	account_id	amount	date
978	DBT	103	$100.00	2004-01-22
979	CDT	103	$25.00	2004-02-05
980	DBT	104	$250.00	2004-03-09
981	DBT	105	$1000.00	2004-03-25
982	CDT	105	$138.50	2004-04-02
983	CDT	105	$77.86	2004-04-04
984	DBT	106	$500.00	2004-03-27

SQL

- SQL (pronounced S-Q-L or SEQUEL) is a language designed to query relational databases
- Used by most financial and commercial companies
- The result of an SQL query is always a table
- It's a nonprocedural language: define inputs and outputs; how the statement is executed is left to the *optimizer*
- How long SQL queries depends on optimization that is opaque to user (which is great!)
- SQL is a language that works with many types of databases:
  - MySQL, SQLite, Hive, BigQuery (Google), Presto (Meta), Redshift (Amazon), ...
  - Performance will vary, but generally faster than standard data frame manipulation in R (and much more scalable)

Components of a SQL query

- SELECT columns
- FROM a table in a database
- WHERE rows meet a condition
- GROUP BY values of a column
- ORDER BY values of a column when displaying results
- LIMIT to only X number of rows in resulting table

- Always required: SELECT and FROM. Rest are optional.
- You may recognize some of the syntax because the `dplyr` package in R is inspired by SQL logic

Aggregate functions

- SELECT can be combined with functions such as SUM, COUNT, AVG... when using GROUP BY

```
SELECT
  account_id,
  SUM(amount) AS total_amount,
  COUNT(*) AS n_transactions
FROM transactions
GROUP BY account_id
```

- COUNT(*) will count the number of rows and store it as a new column called `n_transactions`

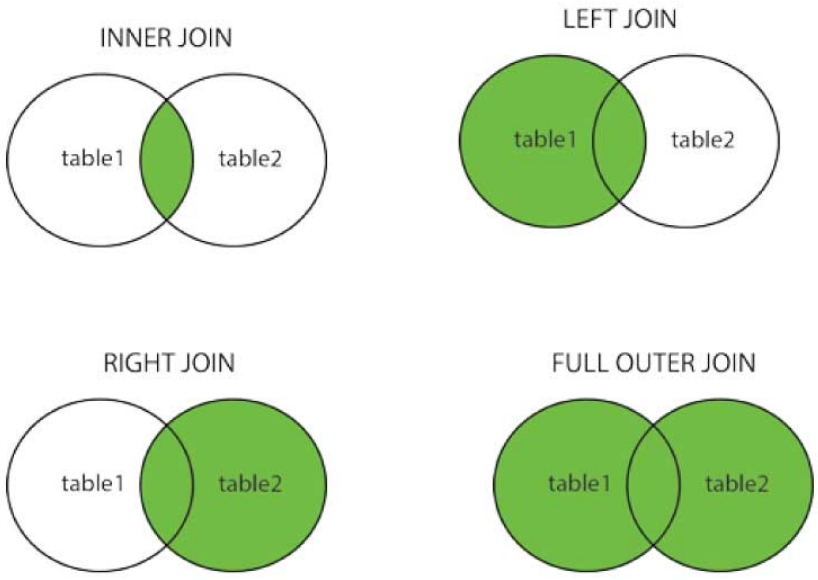
JOINs

- Key advantage of SQL: easy to merge multiple tables using JOIN based ON a common key.

```
SELECT
 a.cust_id,
 SUM(b.amount) AS total_amount,
FROM accounts a
JOIN transactions b
  ON a.account_id = b.account_id
```

Types of JOINs

- Default is INNER JOINs: only merged data with matched keys is kept in output table
- Other options: LEFT JOIN, RIGHT JOIN, FULL OUTER JOIN
- Values of rows that are not matched are placed with NULLs



SQL vs R's `dplyr`

SQL	**`dplyr`**
SELECT	select, rename, mutate, summarise
FROM	<implicit>
WHERE	filter
HAVING	filter
GROUP BY	group_by
ORDER BY	arrange
{X}_JOIN	{x}_join
UNION ALL	bind_rows

SQL at scale

Google BigQuery

- One of many commercial SQL databases available (Amazon RedShift, Microsoft Azure, Oracle Live SQL...)
- Used by many financial and commercial companies
- **Advantages:**
  - Easy to set up; I can give you access
  - Integration with other Google data storage solutions (Google Drive, Google Cloud Storage)
  - Scalable: same SQL syntax for datasets of *any* size
  - Easy to collaborate and export results
  - Affordable pricing and cost control
  - API access allows integration with R or python
  - Excellent documentation