Relational Model

Uses **tables** for data & relations between data

Usually employs **SQL**

A Sample Relational Database

Attributes

customer_id	*customer_name*	*customer_street*	*customer_city*
192-83-7465	Johnson	12 Alma St.	Palo Alto
677-89-9011	Hayes	3 Main St.	Harrison
182-73-6091	Turner	123 Putnam Ave.	Stamford
321-12-3123	Jones	100 Main St.	Harrison
336-66-9999	Lindsay	175 Park Ave.	Pittsfield
019-28-3746	Smith	72 North St.	Rye

Records

(a) The *customer* table

A table: multiple columns

A column: unique name

account_number	*balance*
A-101	500
A-215	700
A-102	400
A-305	350
A-201	900
A-217	750
A-222	700

(b) The *account* table

customer_id	*account_number*
192-83-7465	A-101
192-83-7465	A-201
019-28-3746	A-215
677-89-9011	A-102
182-73-6091	A-305
321-12-3123	A-217
336-66-9999	A-222
019-28-3746	A-201

(c) The *depositor* table

Relational Model

- Bad schema

Duplication

customer_id	*customer_name*	*customer_street*	*customer_city*	*account_number*
192-83-7465	Johnson	12 Alma St.	Palo Alto	A-101
192-83-7465	Johnson	12 Alma St.	Palo Alto	A-201
677-89-9011	Hayes	3 Main St.	Harrison	A-102
182-73-6091	Turner	123 Putnam St.	Stamford	A-305
321-12-3123	Jones	100 Main St.	Harrison	A-217
336-66-9999	Lindsay	175 Park Ave.	Pittsfield	A-222
019-28-3746	Smith	72 North St.	Rye	A-201

- Tables may be stored in files
  - Relational model hides such low-level implementation details

SQL

- SQL: widely used non-procedural language
- **Input**: set of tables + **Constraints** -------> **Output**: 1 table

SQL Query Example I

Find the name of the customer with customer-id 192-83-7465:

customer_id	*customer_name*	*customer_street*	*customer_city*
192-83-7465	Johnson	12 Alma St.	Palo Alto
677-89-9011	Hayes	3 Main St.	Harrison
182-73-6091	Turner	123 Putnam Ave.	Stamford
321-12-3123	Jones	100 Main St.	Harrison
336-66-9999	Lindsay	175 Park Ave.	Pittsfield
019-28-3746	Smith	72 North St.	Rye

select *customer.customer_name*
from *customer*
where *customer.customer_id* = '192-83-7465'

output

input

constraints

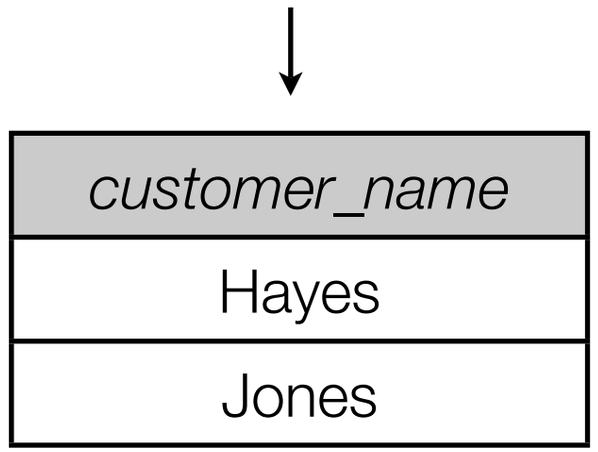
customer_name
Johnson

SQL Query Example II

Find all customers living in Harrison

customer_id	*customer_name*	*customer_street*	*customer_city*
192-83-7465	Johnson	12 Alma St.	Palo Alto
677-89-9011	Hayes	3 Main St.	Harrison
182-73-6091	Turner	123 Putnam Ave.	Stamford
321-12-3123	Jones	100 Main St.	Harrison
336-66-9999	Lindsay	175 Park Ave.	Pittsfield
019-28-3746	Smith	72 North St.	Rye

select *customer.customer_name*
from *customer*
where *customer.customer_city* = ‘Harrison’



customer_name
Hayes
Jones

SQL Query Example III

Find the balances of all accounts held by the customer with customer-id 192-83-7465

account_number	*balance*
A-101	500
A-215	700
A-102	400
A-305	350
A-201	900
A-217	750
A-222	700

(b) The *account* table

customer_id	*account_number*
192-83-7465	A-101
192-83-7465	A-201
019-28-3746	A-215
677-89-9011	A-102
182-73-6091	A-305
321-12-3123	A-217
336-66-9999	A-222
019-28-3746	A-201

(c) The *depositor* table

select *account. account_number, account. balance*
from *depositor , account*
where *depositor.customer_id* = ‘192-83-7465’ and
depositor.account_number = account.account_number

↓

account_number	*balance*
A-101	500
A-201	900

SQL DDL

- Provides a rich DDL

create table *account*
(*account_number* **char**(10)*,*
balance **integer**)

  - creates the *account* table
    - updates data dictionary
- Application programs
  - written in *host* language: C++, Java
  - embeds SQL queries to access data
- Language provides API to send DDL/DML to DB
  - ODBC, JDBC