Some more terminology…

- Schema
  - The structure that contains descriptions of objects created by a user, such as base tables, views, and constraints, as part of a database.
- Catalog
  - A set of schemas that, when put together, constitute a description of a database.

DALHOUSIE UNIVERSITY
Inspiring Minds

Schemas

- External data model
  - The view of users of the database
    - Some users may operate through a *database view* and not see all data.
- Conceptual schema
  - A detailed, technology-independent specification of the overall structure of the organizational data.
    - Covers all external views of the data.
- Internal schema
  - Logical schema
    - The representation of a database for a particular data management technology
  - Physical schema
    - Specifications for how data from a logical schema are stored in a computer's secondary memory by a DBMS

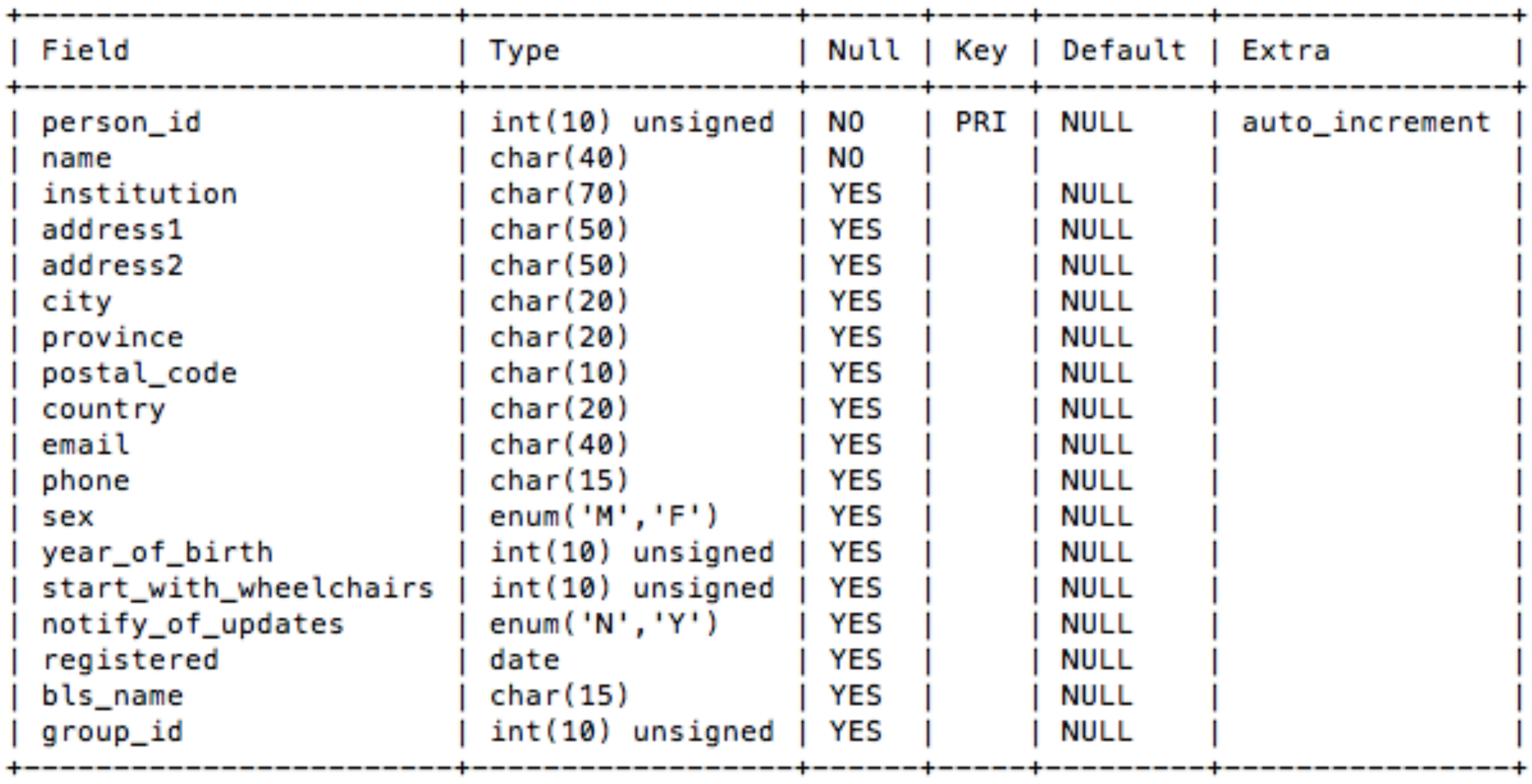
Schemas – Relational Databases

- Relations are stored in rows of tables.
- Entity Relationships are represented by two rows in different tables that share a common column value.

- The schema includes a description of the tables, their columns, and the data types of the columns.

DALHOUSIE UNIVERSITY *Inspiring Minds*

Example table description - mysql



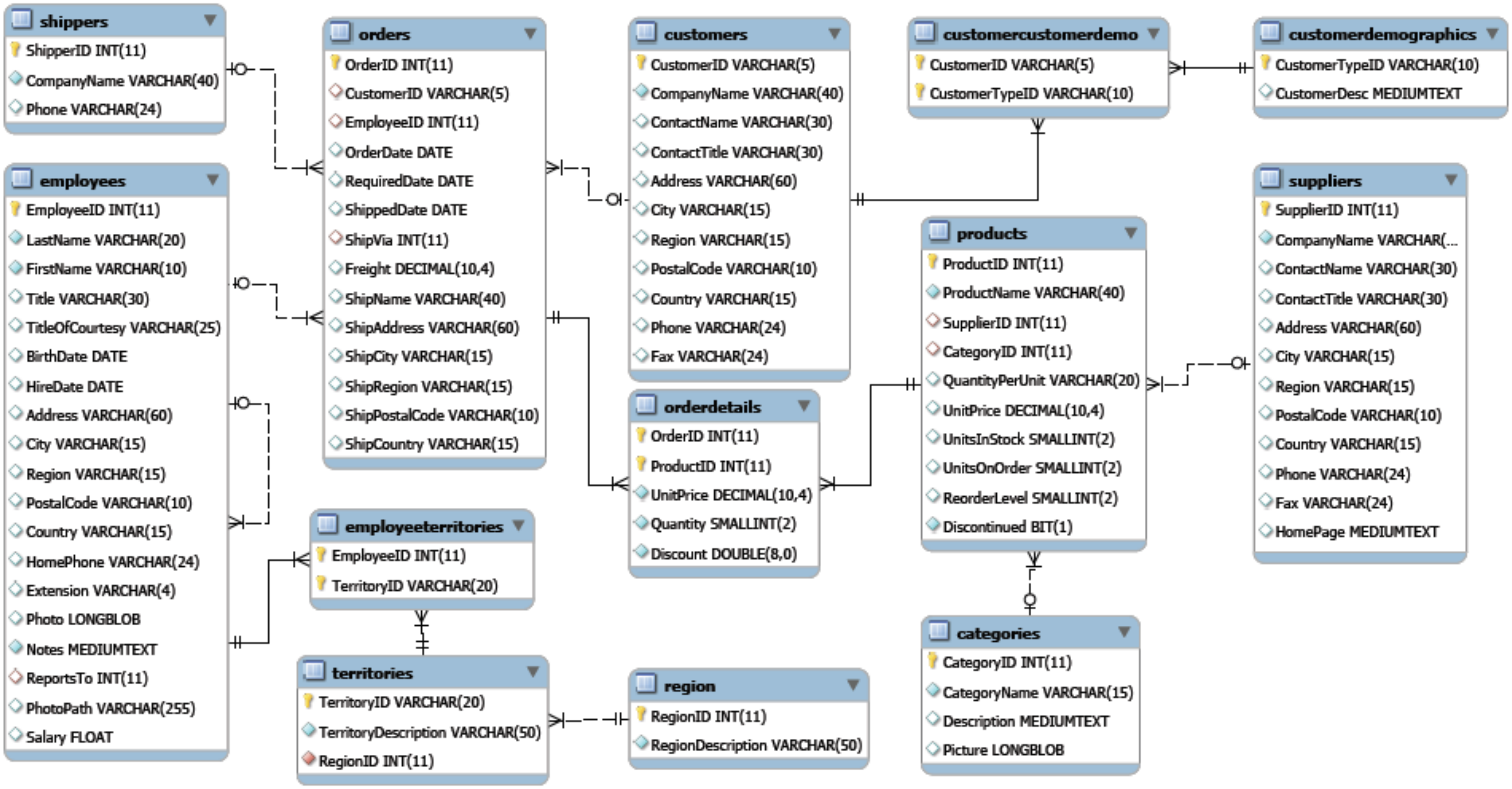
```
+------------------------+------------------+------+-----+---------+----------------+
| Field                  | Type             | Null | Key | Default | Extra          |
+------------------------+------------------+------+-----+---------+----------------+
| person_id              | int(10) unsigned | NO   | PRI | NULL    | auto_increment |
| name                   | char(40)         | NO   |     |         |                |
| institution            | char(70)         | YES  |     | NULL    |                |
| address1               | char(50)         | YES  |     | NULL    |                |
| address2               | char(50)         | YES  |     | NULL    |                |
| city                   | char(20)         | YES  |     | NULL    |                |
| province               | char(20)         | YES  |     | NULL    |                |
| postal_code            | char(10)         | YES  |     | NULL    |                |
| country                | char(20)         | YES  |     | NULL    |                |
| email                  | char(40)         | YES  |     | NULL    |                |
| phone                  | char(15)         | YES  |     | NULL    |                |
| sex                    | enum('M','F')    | YES  |     | NULL    |                |
| year_of_birth          | int(10) unsigned | YES  |     | NULL    |                |
| start_with_wheelchairs | int(10) unsigned | YES  |     | NULL    |                |
| notify_of_updates      | enum('N','Y')    | YES  |     | NULL    |                |
| registered             | date             | YES  |     | NULL    |                |
| bls_name               | char(15)         | YES  |     | NULL    |                |
| group_id               | int(10) unsigned | YES  |     | NULL    |                |
+------------------------+------------------+------+-----+---------+----------------+
```

Mysql data types

- Numeric
  - Int (4 bytes)
  - Tinyint (1 byte)
  - Smallint (2 bytes)
  - Mediumint (3 bytes)
  - Bigint (8 bytes)
  - Float
  - Double
- Enum
- Set
- Date and time
  - Date
  - Datetime
  - Timestamp
  - Time
  - Year
- String
  - Char
  - Varchar
  - Blob / Text
  - Tinyblob
  - Smallblob
  - Mediumblob
  - Longblob

DALHOUSIE UNIVERSITY *Inspiring Minds*

Database lab



Sample database in csci3901 from
http://www.zentut.com/sql-tutorial/sql-sample-database/

DALHOUSIE UNIVERSITY *Inspiring Minds*

Basic SQL operations

- Insert
- Query
  - “select” statement
- Delete
- Update

Insert basics

- Insert into <table> (<column list>) values <tuples>
- Omit (<column list>) when specifying all values
  - Insert into person values (NULL, "Jack", 30, 20000), (NULL, "Kathy", 28, 25000);
- Include <column list> if using the default values for all other columns
  - Insert into person (name, age, salary) values ("Jack", 30, 20000), ("Kathy", 28, 25000);

DALHOUSIE UNIVERSITY
Inspiring Minds

Query basics

- Focus on basic set operations
  - Set restriction with a predicate
    - Structure of a single “select” command
  - Typed set union
    - Joining of the outputs of two “select” commands
  - Typed set intersection
    - Joining of the outputs of two “select” commands
  - Typed set difference
    - Joining of the outputs of two “select” commands

Set restriction with a predicate

Basic select statement

- Select <column list> from <table> where <column criteria>;

  Output   Input   Predicate

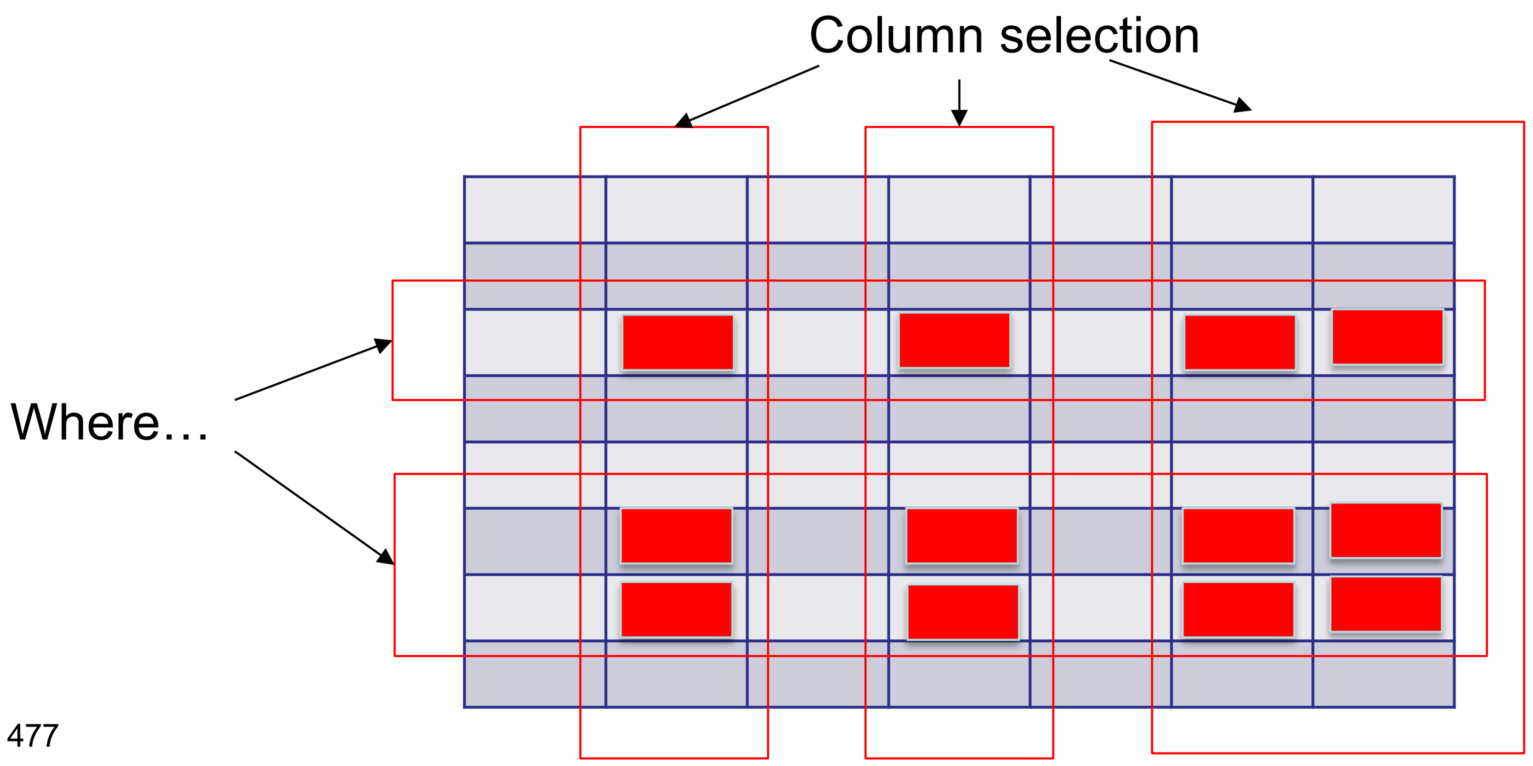
- Example:

  select person_id, name, e-mail from person
    where name = " Mike " and city = " Canmore ";

  select * from person where city = " Halifax ";

Basic select statement

- Use a proposition to identify which elements to select from the set
- Use a list of columns to identify what data to report from that selection



Select "from" element – input specification

- Identify the source set
  - One table
  - Multiple tables
    - Use all row combinations of the multiple tables
    - Called "joins"
      - 4 variants for later: inner join, outer join, left join, right join
    - For tables a, b, c creates the set a X b X c
  - Fabricated tables from subqueries
    - Use the output of one SQL query as the input table for another query
    - More on subqueries later
- Create short names / aliases for tables
  - Useful for duplicated tables or fabricated tables

DALHOUSIE UNIVERSITY
Inspiring Minds

Select “from” element

- Examples:

… from person …	single table alone
… from person as p …	single table with alias
… from person, courses …	two tables
… from person as p, courses as c …	two tables with aliases

Select column list – output specification

- Identifies what to return from the query
- Could be
  - A list of column names
    - Just the name, if unique
    - TableName.ColumnName or TableAlias.ColumnName if not unique
  - *
    - Specifies all table columns
    - Can be TableName.* or TableAlias.*
  - Transformations of columns
  - Added keywords
    - Eg. DISTINCT
- Can name outgoing columns

Select column list

- Examples

  select name, age from person where …

  select * from person where …

  select name as Full_Name from person where …

  select person.name, course.name as course from person, course where …

DALHOUSIE UNIVERSITY
Inspiring Minds

Select column list - transformations

- Avg()
- Count()
- Min()
- Max()
- Std()
- Variance()
- Sum()
- Format()

DALHOUSIE UNIVERSITY
Inspiring Minds

Select column list - transformations

- Concat()
- Lcase() or lower()
- Ucase() or upper()
- Left(), Right, or Mid()
- Length()
- Ltrim(), Rtrim(), Trim()
- Lpad() or Rpad()
- …

DALHOUSIE UNIVERSITY *Inspiring Minds*

Select transformations

- Examples

```
select count( name ) from person where …
select avg( age ) from person where …
select concat( name, " – ", age ) from person where …
select sum( fees ) from registration where …
select max( salary ) from person where …
```

Select “where” – selection predicate

- Identifies which rows to keep from the input
- Uses
  - Maintain the relation between tables
    - Where person.person_id = registration.person_id
  - Select particular elements
    - Where name = " Doug "
- Allows for Boolean operators
  - …. And ….
  - …. Or ….
  - Not ….
  - Use parentheses to help with the Boolean logic

Selection predicates

- Standard comparators
  - =, !=, <>, >, <, >=, =<, !<, !>
- Numeric ranges – “between”
  - Select name from person where salary between 32000 and 50000
- Set inclusion – “in”
  - Select person_id from registration where course_id in (1, 2, 3)
  - Select distinct person_id from registration where course_id in (1, 2, 3)
- Near matches – “like”
  - % matches 0 or more characters, _ matches 1 character
  - Select name from person where name like “C%”
  - Works on numbers too: select * from person where salary like “3%”
- NULL check – “is null”

Additional “select” specifications

- Order by <column list> [ASC | DESC]
  - Allows you to sort the data
- Group by <column list>
  - Collects similar records for aggregation transformations like count or sum
- Group by <column list> having <clause>
  - Like “group by” but lets you select a subset of groups
- Limit n
  - Report only the first n records
  - “limit” for mysql, “top” for some other systems
- Distinct
  - Only provide unique rows of output
  - Duplication can happen when you’re reporting a subset of columns