Lab #10.1: Form SQL queries

Overview

Objective

The objective of this lab is to form SQL queries to search the database.

Outcomes

- Create SQL queries against one or more tables.
- Use `psql` output functionality to facilitate data management.

Knowledge expected

- Form SQL queries against one or more tables:
    - Display all records.
    - Display sorted records.
    - Display all records that match a condition.
    - Display all records that match a compound condition.
- To use the `psql \w` sub-command to save SQL *statements* to a file.
- To use the `psql \g` sub-command to save SQL query *results* to a file.

Submission instructions

- **READ ALL THE WORDS**
- **You must follow ALL submission instructions**: Submission instructions are explained in the "Lab 10 submission details" document, posted on BrightSpace. ANY submission instructions below NOT followed result in a grade of zero.
  Note: This includes extra functionality that has not been requested.
- You are expected to complete all exercises, even those not are not required to be submitted.

Background

SQL syntax

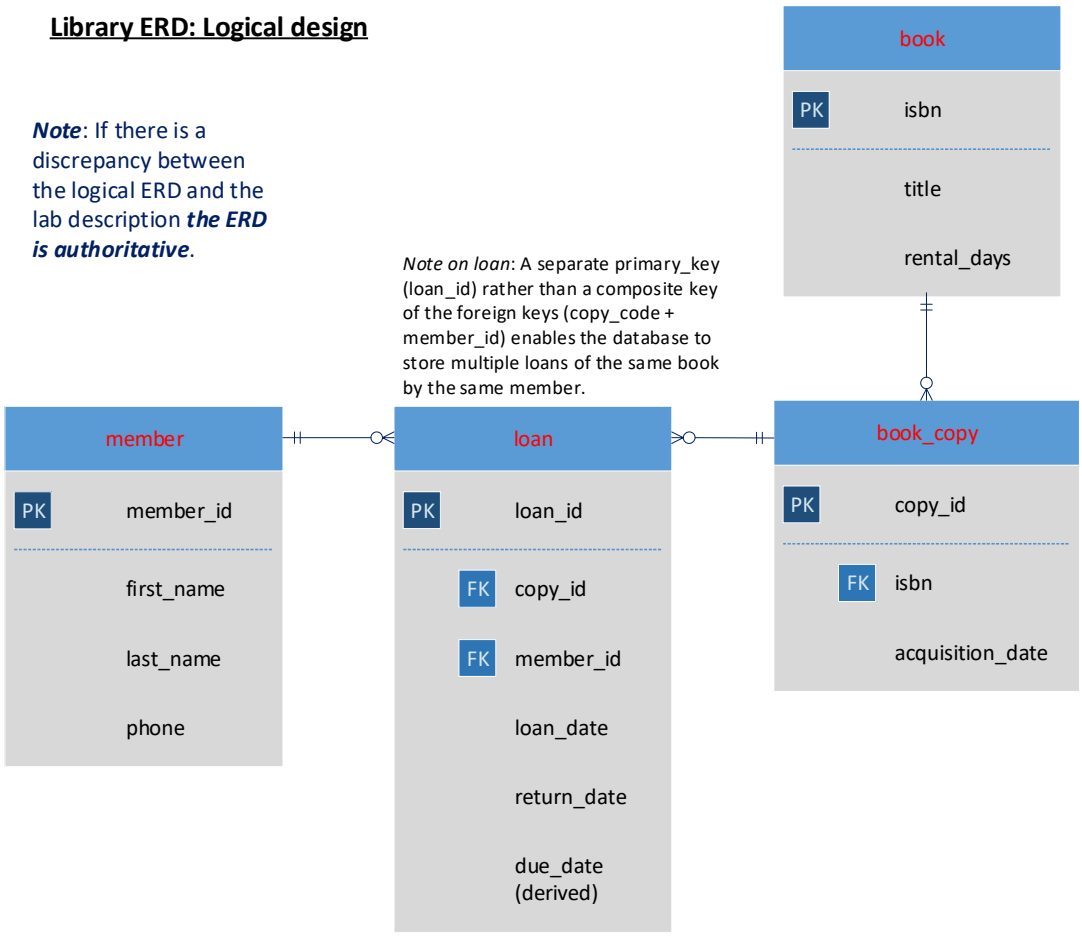
- The semicolon at the end of the statement is part of the SQL syntax.
- SQL commands are case-insensitive.

Common syntax failures

- *Example* of a regular prompt for a superuser: `dabase_name=#`

- *Example* of a regular prompt for a non-superuser: `dabase_name=>`
- *Example* of prompt indicating a missing closing bracket: `dabase_name(#`
- *Example* of prompt indicating a missing closing semi-colon: `dabase_name-#`
- To cancel a bad command use: `\r`.

Scenario: Library logical design



Section A – Form SQL queries on a single table

Note: The query examples and exercises given below may not reflect your data. Adjust where necessary to achieve successful queries.

***Syntax practice #1*: Display all records.**

- Select all records.
  - ***Syntax***: `SELECT * FROM table;`
  - ***Example***: `SELECT * FROM book;`

- Select specified attributes for all records.
  Note: To display only certain attributes, replace the '*' with a comma-separated list of column names.
    - ***Syntax***: `SELECT column(s) FROM table;`
    - ***Example***: `SELECT first_name, last_name FROM member;`
    - ***Exercise***: Display the book title and rental days of all records in the `book` table.

***Syntax practice #2*: Display all records, sorted.**

- Sort all records: `ORDER BY`
    - ***Syntax***: `SELECT * FROM table ORDER BY column(s) [ASC|DESC];`
    - ***Example***: `SELECT * FROM member ORDER BY last_name ASC, first_name ASC;`
    - ***Exercise***: Display the book title and ISBN number of all books sorted by title in ascending order.
    - ***Exercise***: Display first and last name of all members sorted by last name in ascending order.

***Syntax practice #3*: Display all records that match a condition using comparison operators.**

We are selecting all records that match a condition using the `WHERE` clause.

The following comparison operators are accepted by the `WHERE` clause: =, !=, <, >, <=, >=

- Select records using the `WHERE` clause with a comparison operator.
    - ***Syntax***: `SELECT * FROM table WHERE column = value;`
    - ***Example***: `SELECT * FROM book WHERE title = 'Moby Dick';`
    - ***Exercise***: Display the first and last name of all member records where the last name is *not* 'Simpson'.
    - ***Exercise***: Display the title and rental days of all books where the number of rental days is greater than 7 days.
    - ***Exercise***: Select all book copies where the acquisition date is less or equal than 2022-12-31 (or any other reasonable date).

***Syntax practice #4*: Select all records that match a condition using alternative match criteria.**

We are selecting all records that match a condition using the `WHERE` clause with alternative match criteria.

- `BETWEEN AND` (equivalent to >= and <=)
    - ***Example***: `SELECT * FROM book_copy WHERE acquisition_date BETWEEN '2010-12-31' AND '2020-12-31';`
- `LIKE #` *Note*: resource intensive query
    - ***Example***: `SELECT * FROM member WHERE last_name LIKE 'Chap%';`
    - ***Example***: `SELECT * FROM member WHERE last_name LIKE '%eade';`
- `IS [NOT] NULL`

    - ***Example***: `SELECT * FROM loan WHERE return_date IS NOT NULL;`
- `IN`
    - ***Example***: `SELECT * FROM member WHERE last_name IN ('Reader', 'Digest', 'Noname');`

***Syntax practice #5*: Select all records that match a compound condition.**

- `AND`
    - ***Example***: `SELECT * FROM member where last_name = 'Reader' AND first_name = 'Robin';`
- `OR`
    - ***Example***: `SELECT * FROM member where last_name = 'Reader' OR first_name = 'Chris';`

Section B – Save a query to a file

The interactive `psql` utility has sub-commands to save SQL statements written at the `psql` prompt and query results to a file in the current directory.

***Syntax practice #6*: Save a query.**

- ***Syntax***: **`\w file`** # saves the last executed SQL command to a file from the query buffer. *Note*: To view the query buffer use `\p`.
- ***Exercise***:
    - In `psql` display all books by title only.
    - Save the query to a file named `book_title.query` using the `psql` sub-command `\w`.
    - Verify that the query command has been saved in a text file in your current Linux directory.

***FYI*: Save a query result.**

To save the query *result* of a previously executed SQL query to a file use the `psql` subcommand: `\g file`.

Section C – SQL Queries: Join multiple tables

Overview

- There four types of joins: inner join, full join, left join, right join. The most common join type is the inner join.
- Tables are joined on a common column provided by the primary key in the parent table that is referenced by the foreign key in the related table.
- The basic join syntax identifies:

- the attributes to display
- the tables to join
- the common field in the table-pairs to join: PK-FK
- optional clauses: match, sort

Join two tables

Syntax example for joining two tables

```
SELECT column(s)
  FROM table1
  JOIN table2 ON table1.pkey = table2.fkey
  [WHERE condition]
  [ORDER BY];
```

Note: Attributes (columns) may include the table name. This must be used when we need to disambiguate the same column name in different tables.
Example: `SELECT member.last_name, author.last_name`

***Syntax practice #7*: Join two tables and sort records.**

- ***Request***: Display all book copies for all books, sort by title.
- ***Query***:

```
SELECT copy_id, title
  FROM book
  JOIN book_copy ON book.isbn = book_copy.isbn
  ORDER BY title;
```

***Syntax practice #8*: Join two tables and filter result set based on condition.**

- ***Request***: Display all book copies for the book 'Moby Dick'.
- ***Query***:

```
SELECT title, copy_id
  FROM book
  JOIN book_copy ON book.isbn = book_copy.isbn
  WHERE title = 'Moby Dick';
```

***Exercise #9*: Join two tables, sort result set.**

- ***Request***: Display title and acquisition date of all book copies & sort by book title.
  - *Which attributes do you list for display?*
  - *Which tables do you join?*
  - *Which column(s) do you use to join the tables?*
  - *Which optional clause(s) do you include?*

***Exercise #10*: Join two tables and filter result set based on compound condition.**

- ***Request***: Display loan date, return date and copy id for all loans (past & current) by the member "Chris Chapter".
    - *Which attributes do you list for display?*
    - *Which tables do you join?*
    - *Which column(s) do you use to join the tables?*
    - *Which conditions are used in the matching clause?*

  Note: Display a minimum of two records (add more records where necessary).

Join tables with intersecting table

Syntax example for joining three tables that represent a many-to-many relationship

```
SELECT table1.column, table2.column
  FROM table1
  JOIN linktable ON table1.primarykey = linktable.foreignkey
  JOIN table2    ON table2.primarykey = linktable.foreignkey
```

***Syntax practice #11*: Join three tables that include an intersecting table, filter and sort result set.**

- ***Request***: Display all book copies borrowed by the member "Robin Reader" (past & current).
    - *List attributes to display:* [member.]first_name, [member.]last_name, [book_copy.]isbn, [loan.]loan_date, [loan.]return_date
      Note: Data in brackets are optional but it helps to identify the tables that have to be included in the join.
    - *List tables to join:* member, book_copy, loan
    - *Identify the column(s) to join the tables:*
        - member & loan: member_id
        - book_copy & loan: copy_id
    - *Specify required clause(s):* compound matching clause to filter for the member "Robin Reader"

  Note: Display a minimum of two records (add more records where necessary).
- ***Query***:

```
SELECT first_name, last_name, isbn, loan_date, return_date
  FROM book_copy
  JOIN loan   ON book_copy.copy_id = loan.copy_id
  JOIN member ON member.member_id  = loan.member_id
  WHERE last_name = 'Reader' AND first_name = 'Robin';
```

  Note: The table in the query order can be reversed, starting with the member table.

Join multiple tables

***Syntax practice #12*: Join three tables & sort result set.**

- ***Request***: Display all book loans (current & past) and sort by title.
    - *List attributes to display:* title, loan_date, return_date

- *List tables to join:* _____
- *Identify the column(s) to join the tables:*
  - _____
  - _____
- *Specify required clause(s):* _____

- ***Query***:

```
SELECT title, loan_date, return_date
  FROM book
  JOIN book_copy ON book.isbn         = book_copy.isbn
  JOIN loan      ON book_copy.copy_id = loan.copy_id
  ORDER BY title;
```

***Syntax practice #13*: Join three tables with derived attribute.**

- ***Request***: Display due dates for all book loans (current & past).
  - *List attributes to display:* title, loan_date, due_date (derived from loan_date and rental_days)
  - *List tables to join:* book, book_copy, loan
  - *Identify the column(s) to join the tables:*
    - book, book_copy: isbn
    - book_copy, loan: copy_id
- ***Query***:

```
SELECT title, loan_date, loan_date + rental_days AS due_date
  FROM book
  JOIN book_copy ON book.isbn         = book_copy.isbn
  JOIN loan      ON book_copy.copy_id = loan.copy_id;
```

***Exercise #14*: Join more than three tables & sort result set.**

- ***Request***: Display all book loans (past & current) of all members and sort by last name.
  - *List attributes to display:* last_name, title, loan_date, return_date
  - *List tables to join:* _____
  - *Identify the column(s) to join the tables:*
    - _____
    - _____
    - _____
  - *Specify required clause(s):* _____