

Wilbur Wildcat  
 INFO 579: SQL/NoSQL  
 Summer 2024  
 Week 7 Final Project Report  
 Nayem Rahman, PhD  
 August 16, 2024

## Business Plan

### 1. Come up with a business plan for which you need to collect and store data

The **tackle** database is a central component of the notional ‘Fishing Companion’ app, designed to enhance the angling experience by organizing and managing comprehensive data on baits, rods, reels, and lines. This database is meticulously structured to store detailed information about various tackle configurations, enabling the retrieval of personalized recommendations based on specific fishing conditions. Users can access the most suitable tackle configurations by inputting details such as water type, target species, and weather conditions. The database's robust design ensures that all recommendations are accurate, up-to-date, and tailored to the unique needs of each user. This project underscores the critical importance of the tackle database in storing, managing, and analyzing data, providing precise and valuable insights that are integral to the project's success. The project repository is at <https://github.com/ciiprof0/tackle>.

## Data Sources

### 2. Come up with the data about your business.

The data source is a comprehensive dataset that organizes critical information about various fishing tackle components, including baits, rods, reels, and lines. The dataset is divided into multiple sheets, each focusing on a specific tackle type. The **Tackle** sheet consolidates details such as bait type, presentation method, fish depth, rod type and length, reel gear ratio, and line test strength. The **Baits** sheet provides data on bait types, their effective depth range, and presentation methods. The **Rods** sheet contains information on rod power, action, and length specifications. The **Reels** sheet includes data on reel types and gear ratios, while the **Lines** sheet captures the range of line test strengths across different line types. This structured data, derived from tournament fishing experience, is essential for the Fishing Companion app's database, enabling it to offer accurate and practical recommendations to users. The raw data used to seed the database is at <https://github.com/ciiprof0/tackle/raw/main/data/raw/data.xlsx>.

## Data Models

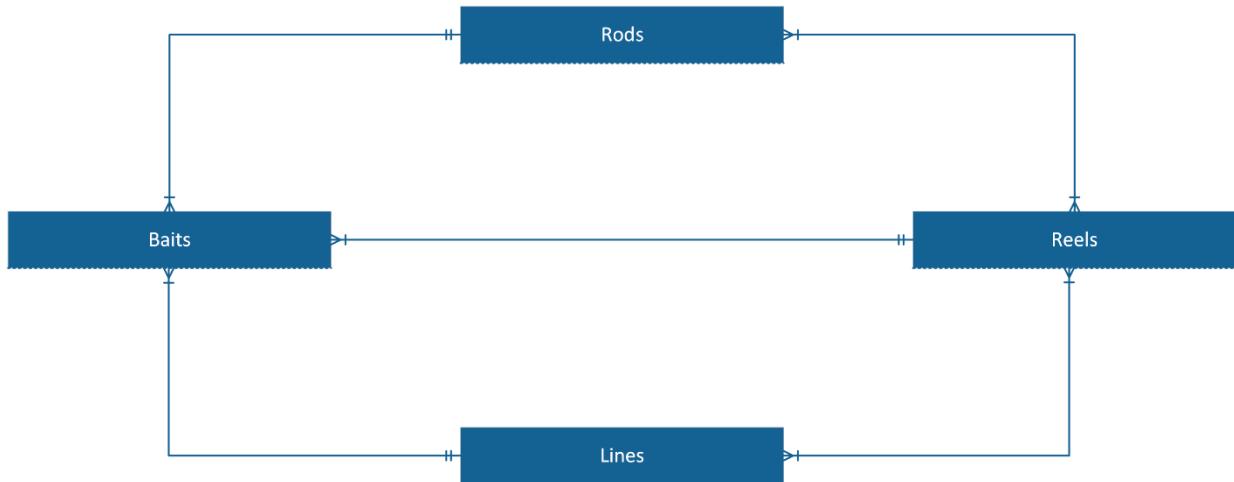
In the relationship between rods and reels, a single rod can be paired with multiple reels, allowing for different combinations of fishing setups. Conversely, a single reel can be used with various rods, offering versatility in utilizing the equipment. Similarly, when considering rods and lines, a single rod can accommodate multiple types of lines and a single line can be applied across different rods, demonstrating a flexible usage pattern. The same principle applies to rods and baits, where a single rod can be used with multiple baits and a single bait can be matched with different rods to suit varying fishing conditions. The `Combos` table is the junction point

that facilitates these many-to-many relationships, linking rods, reels, lines, and baits together in numerous combinations, reflecting the complex interconnections within the fishing gear setup. See Figure 1 for the conceptual model, Figure 2 for the logical model, and Figure 3 for the physical model.

**3. Develop a Conceptual Model. Consider 4 or 5 entities. Make sure you have at least one many-to-many relationship. Explain with data why it's a many-to-many relationship.**

**Figure 1**

*Tackle Database Conceptual Model with Many-to-Many Relationships*

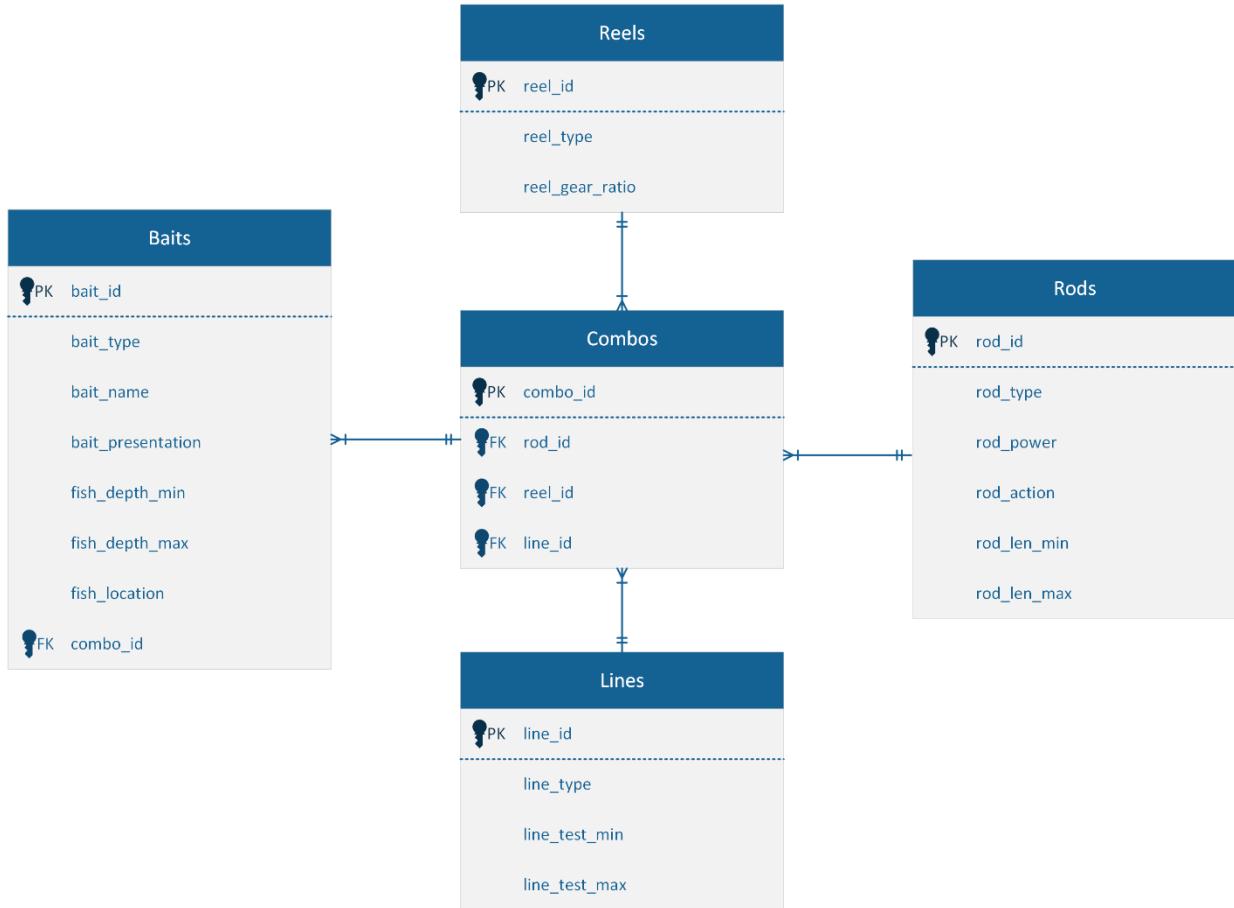


*Note.* Diagram created with Microsoft Visio 365 for the web. Diagram Visio (vsdx), PDF, and PNG files available at <https://github.com/cioprof0/tackle/tree/main/docs/design>

**4. Develop a Logical Model using the Conceptual Model. Make sure you come up with a junction entity to resolve the many-to-many relationship.**

**Figure 2**

*Tackle Database Logical Model*

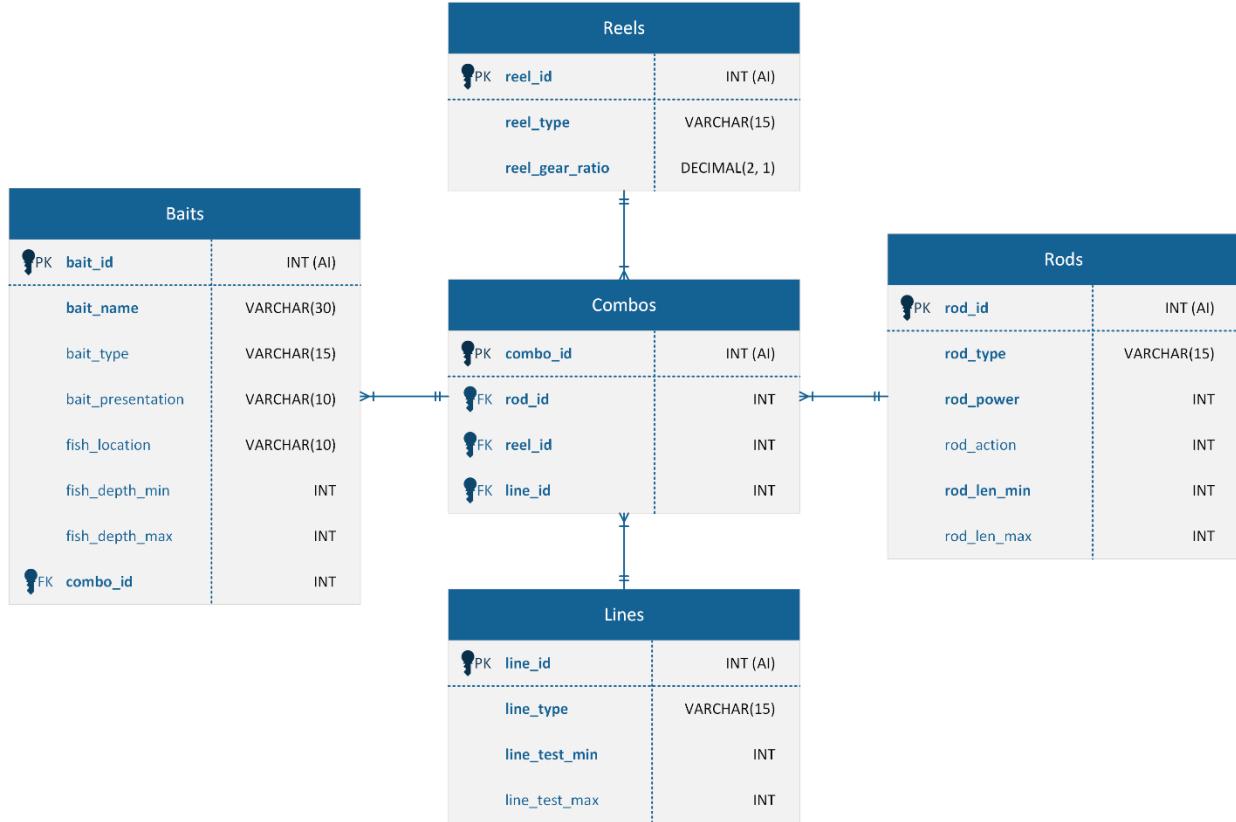


*Note.* Diagram created with Microsoft Visio 365 for the web. Diagram Visio (vsdx), PDF, and PNG files available at <https://github.com/cioprof0/tackle/tree/main/docs/design>

## 5. Develop the physical model based on the Logical Model

**Figure 3**

*Tackle Database Physical Model*



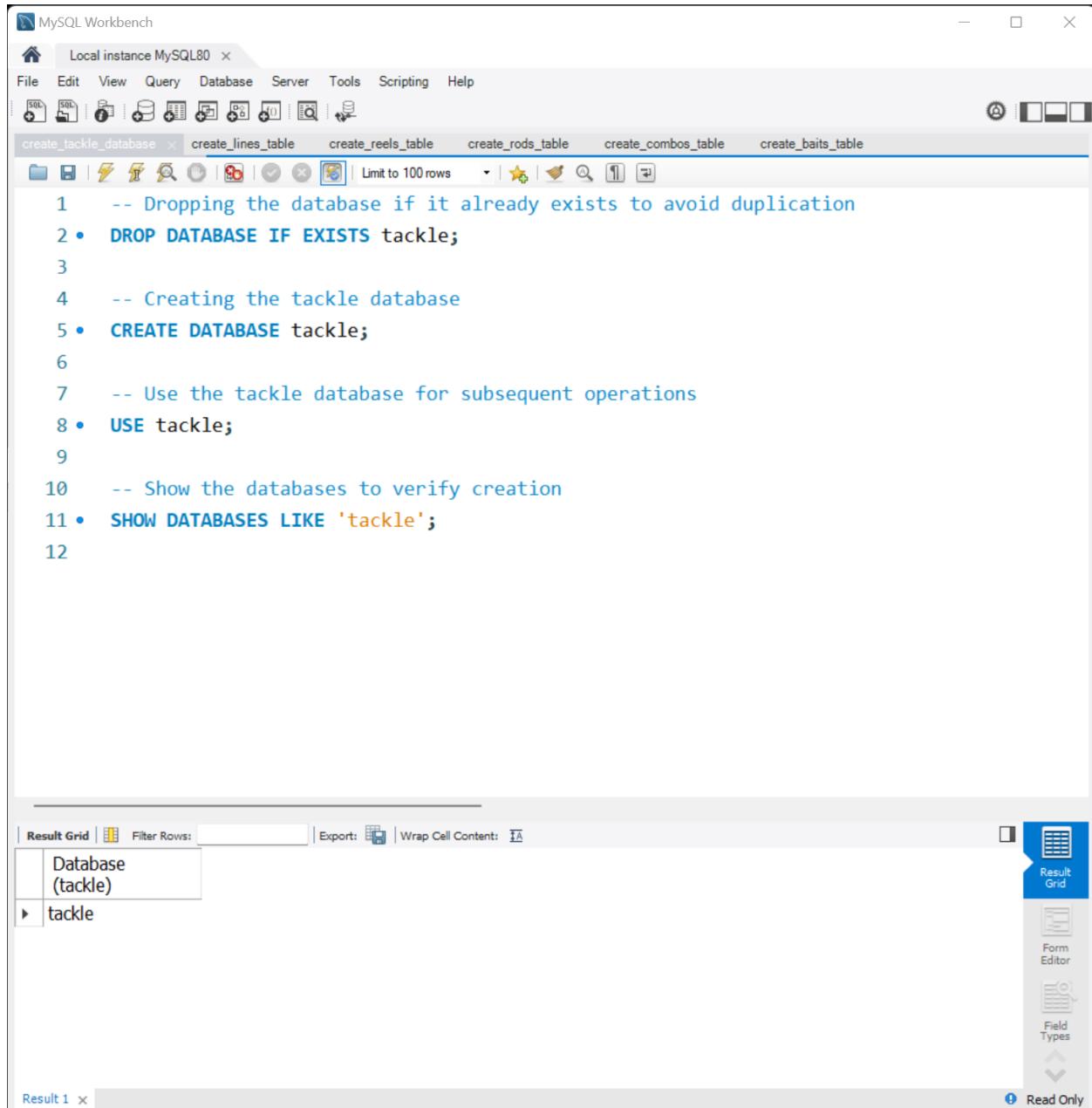
*Note.* NOT NULL attributes are shown in bold font. Diagram created with Microsoft Visio 365 for the web. Diagram Visio (vsdx), PDF, and PNG files available at <https://github.com/cioprof0/tackle/tree/main/docs/design>

## Schema

**Create the database.**

**Figure 4**

*Create the Tackle Database*



The screenshot shows the MySQL Workbench interface with the following details:

- Title Bar:** MySQL Workbench - Local instance MySQL80
- Toolbar:** File, Edit, View, Query, Database, Server, Tools, Scripting, Help.
- Script Editor:** A large text area containing the SQL code for creating the 'tackle' database. The code is as follows:

```

1 -- Dropping the database if it already exists to avoid duplication
2 • DROP DATABASE IF EXISTS tackle;
3
4 -- Creating the tackle database
5 • CREATE DATABASE tackle;
6
7 -- Use the tackle database for subsequent operations
8 • USE tackle;
9
10 -- Show the databases to verify creation
11 • SHOW DATABASES LIKE 'tackle';
12

```

- Result Grid:** A table showing the results of the SHOW DATABASES query. It contains one row with the database name 'tackle'.

Database
(tackle)
tackle

- Status Bar:** Result 1 x Read Only

*Note.* Code at

[https://github.com/ciiprof0/tackle/blob/main/db/schema/setup/create\\_tackle\\_database.sql](https://github.com/ciiprof0/tackle/blob/main/db/schema/setup/create_tackle_database.sql)

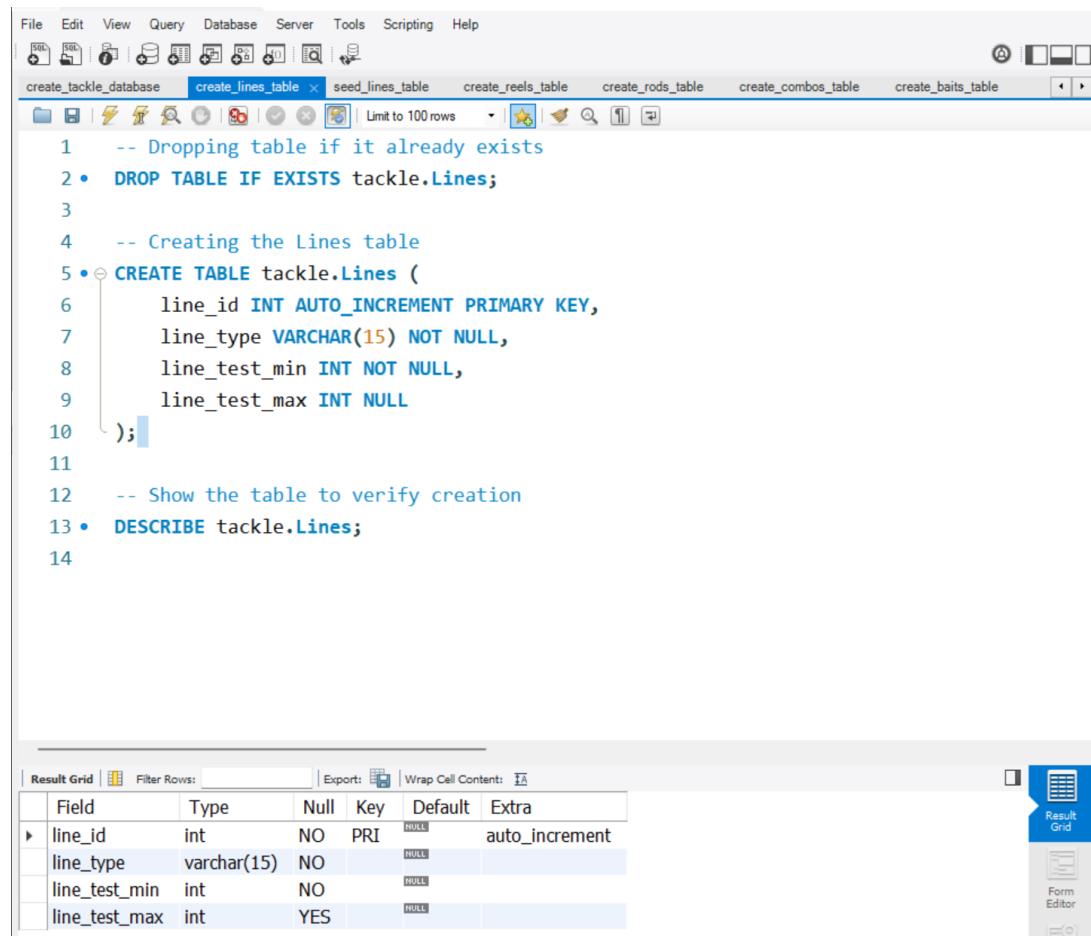
**6. Create tables using a database system. Insert data into the database tables. You must provide the DDL (CREATE TABLE statements), INSERT statements, and SELECT statements.**

*Details: Create the tables that you have come up with (the table must be based on the Physical Model).*

- (a) *Columns, Primary Key (PK), Data Type and length, and NULL/NOT NULL need to be implemented, per the Physical Model.*
- (b) *Show the table definition (DDL) that you implemented (not in a graphical view).*
- (c) *Insert the complete set of data that you have come up with and show the insert statements used.*

**Figure 5**

*Create Lines Table*



The screenshot shows the MySQL Workbench interface with the following details:

- Toolbar:** File, Edit, View, Query, Database, Server, Tools, Scripting, Help.
- Tab Bar:** create\_tackle\_database, **create\_lines\_table** (selected), seed\_lines\_table, create\_reels\_table, create\_rolls\_table, create\_combos\_table, create\_baits\_table.
- Query Editor:**

```

1 -- Dropping table if it already exists
2 • DROP TABLE IF EXISTS tackle.Lines;
3
4 -- Creating the Lines table
5 • CREATE TABLE tackle.Lines (
6     line_id INT AUTO_INCREMENT PRIMARY KEY,
7     line_type VARCHAR(15) NOT NULL,
8     line_test_min INT NOT NULL,
9     line_test_max INT NULL
10 );
11
12 -- Show the table to verify creation
13 • DESCRIBE tackle.Lines;
14

```
- Result Grid:**

Field	Type	Null	Key	Default	Extra
line_id	int	NO	PRI	NULL	auto_increment
line_type	varchar(15)	NO		NULL	
line_test_min	int	NO		NULL	
line_test_max	int	YES		NULL	
- Buttons:** Result Grid, Form Editor.

*Note. Code at*

[https://github.com/ciiprof0/tackle/blob/main/db/schema/tables/create\\_lines\\_table.sql](https://github.com/ciiprof0/tackle/blob/main/db/schema/tables/create_lines_table.sql)

**Figure 6***Seed Lines Table*

The screenshot shows the MySQL Workbench interface with the following details:

- Query Editor:** The current tab is "seed\_lines\_table". The code is as follows:

```

74 • WHERE NOT EXISTS (
75     SELECT 1 FROM tackle.Lines
76     WHERE line_type = 'monofilament' AND line_test_min = 15 AND line_test_max = 20
77 );
78
79 • INSERT INTO tackle.Lines (line_type, line_test_min, line_test_max)
80     SELECT 'monofilament', 12, 20
81 • WHERE NOT EXISTS (
82     SELECT 1 FROM tackle.Lines
83     WHERE line_type = 'monofilament' AND line_test_min = 12 AND line_test_max = 20
84 );
85
86 -- Show the data to verify insertion
87 • SELECT * FROM tackle.Lines;
88

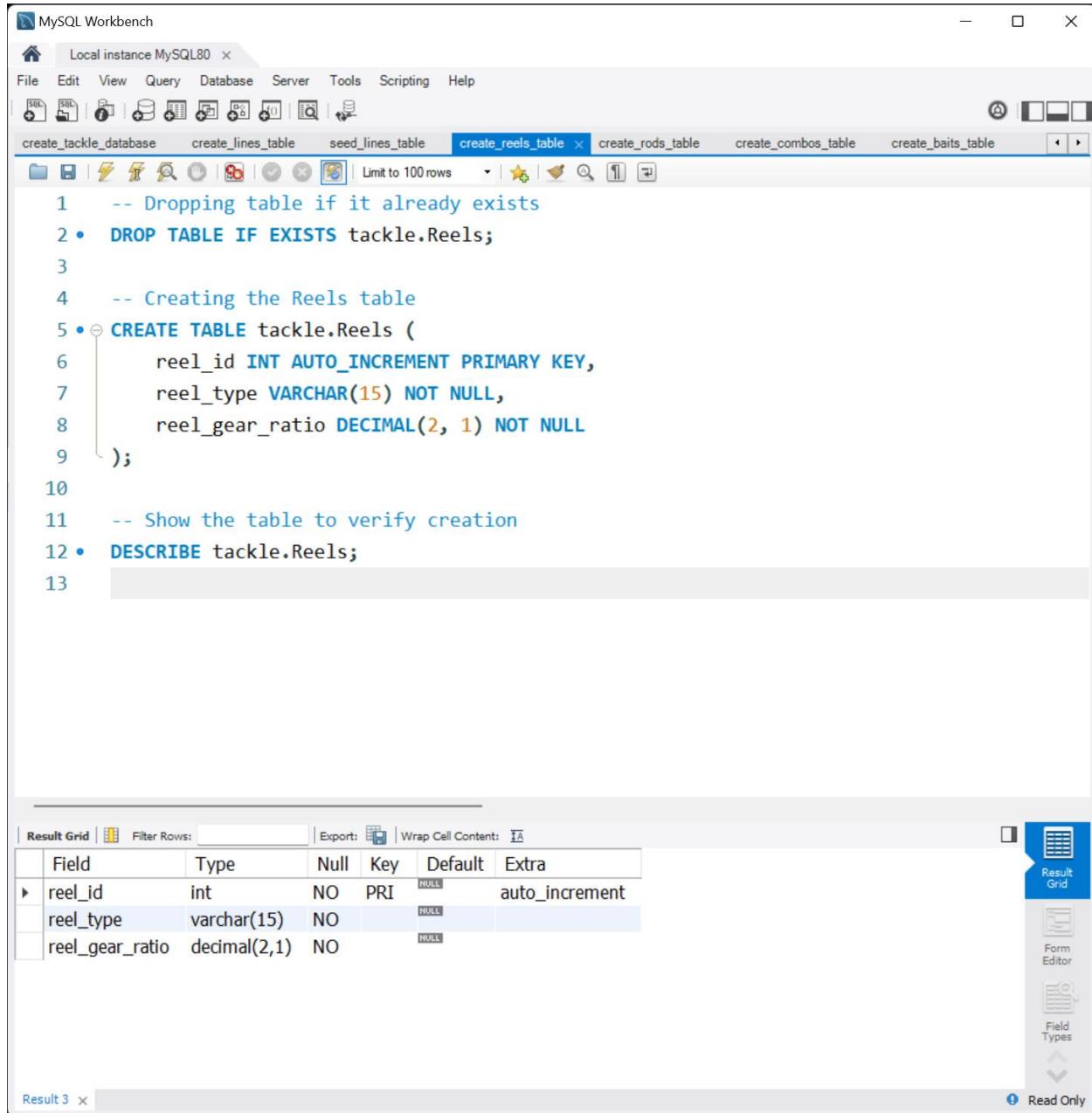
```

- Result Grid:** The results of the final query are displayed in a grid:

line_id	line_type	line_test_min	line_test_max
1	braid	50	65
2	monofilament	10	15
3	fluorocarbon	12	20
4	fluorocarbon	6	10
5	fluorocarbon	6	15
6	fluorocarbon	12	17
7	fluorocarbon	10	15
8	fluorocarbon	20	25
9	fluorocarbon	15	20
10	fluorocarbon	12	25
11	monofilament	15	20
12	monofilament	12	20

- Right Panel:** A sidebar with various tabs: Result Grid (selected), Form Editor, Field Types, Query Stats, and Execution Plan.

*Note.* Code at [https://github.com/ciiprof0/tackle/blob/main/db/seeds/seed\\_lines\\_table.sql](https://github.com/ciiprof0/tackle/blob/main/db/seeds/seed_lines_table.sql)  
 Seed data at <https://github.com/ciiprof0/tackle/raw/main/data/raw/data.xlsx>

**Figure 7***Create Reels Table*


The screenshot shows the MySQL Workbench interface with a query editor window. The title bar says "MySQL Workbench" and "Local instance MySQL80". The menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. The toolbar has various icons for database management. The query editor contains the following SQL code:

```

1 -- Dropping table if it already exists
2 • DROP TABLE IF EXISTS tackle.Reels;
3
4 -- Creating the Reels table
5 • CREATE TABLE tackle.Reels (
6     reel_id INT AUTO_INCREMENT PRIMARY KEY,
7     reel_type VARCHAR(15) NOT NULL,
8     reel_gear_ratio DECIMAL(2, 1) NOT NULL
9 );
10
11 -- Show the table to verify creation
12 • DESCRIBE tackle.Reels;
13

```

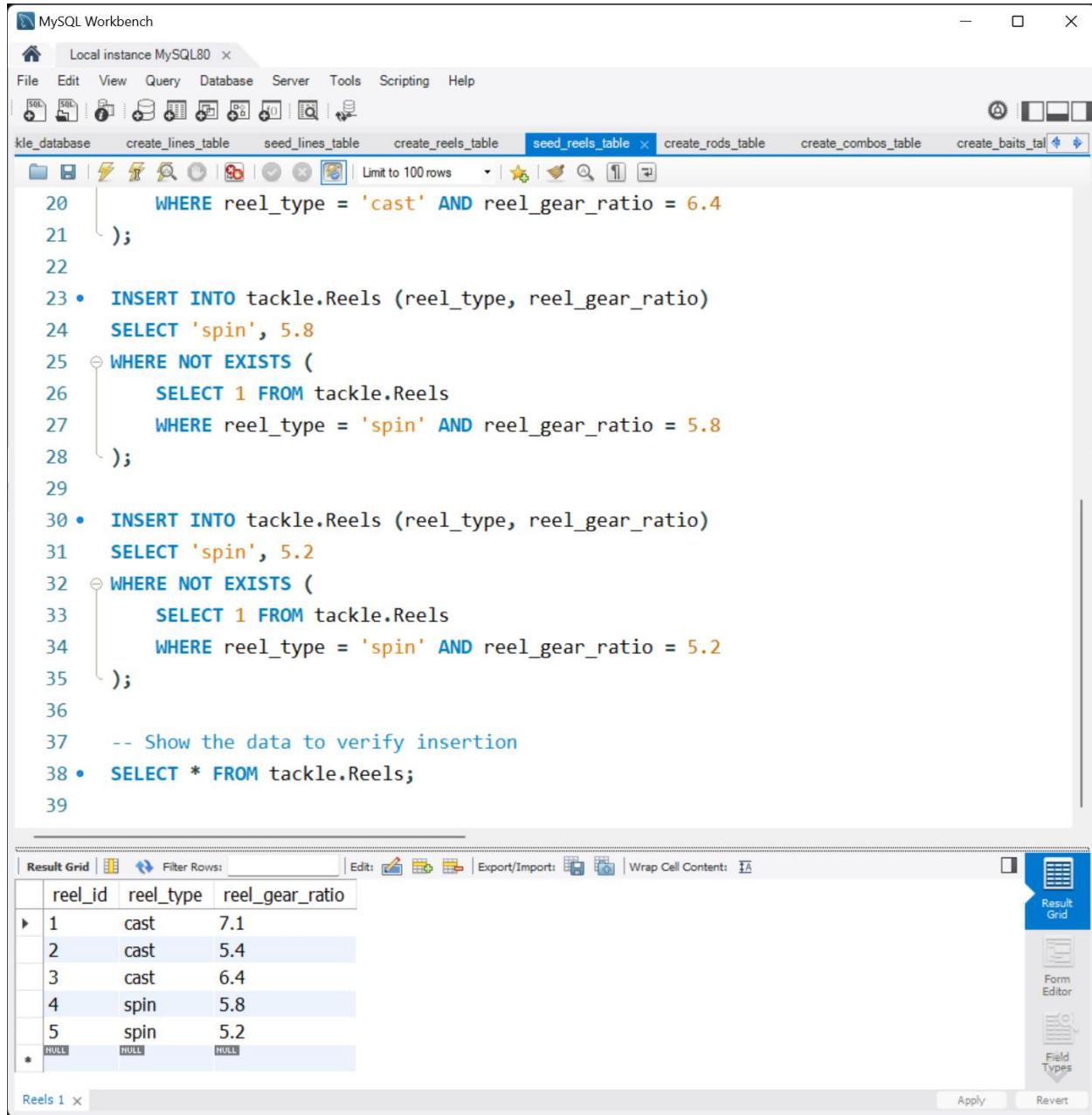
Below the code, the "Result Grid" pane displays the table structure:

Field	Type	Null	Key	Default	Extra
reel_id	int	NO	PRI	NULL	auto_increment
reel_type	varchar(15)	NO		NULL	
reel_gear_ratio	decimal(2,1)	NO		NULL	

The "Result 3" pane at the bottom indicates there are three results. A sidebar on the right shows icons for Result Grid, Form Editor, and Field Types, with "Result Grid" selected.

*Note.* Code at

[https://github.com/cioprof0/tackle/blob/main/db/schema/tables/create\\_reels\\_table.sql](https://github.com/cioprof0/tackle/blob/main/db/schema/tables/create_reels_table.sql)

**Figure 8***Seed Reels Table*


The screenshot shows the MySQL Workbench interface with a query editor and a results grid.

**Query Editor:**

```

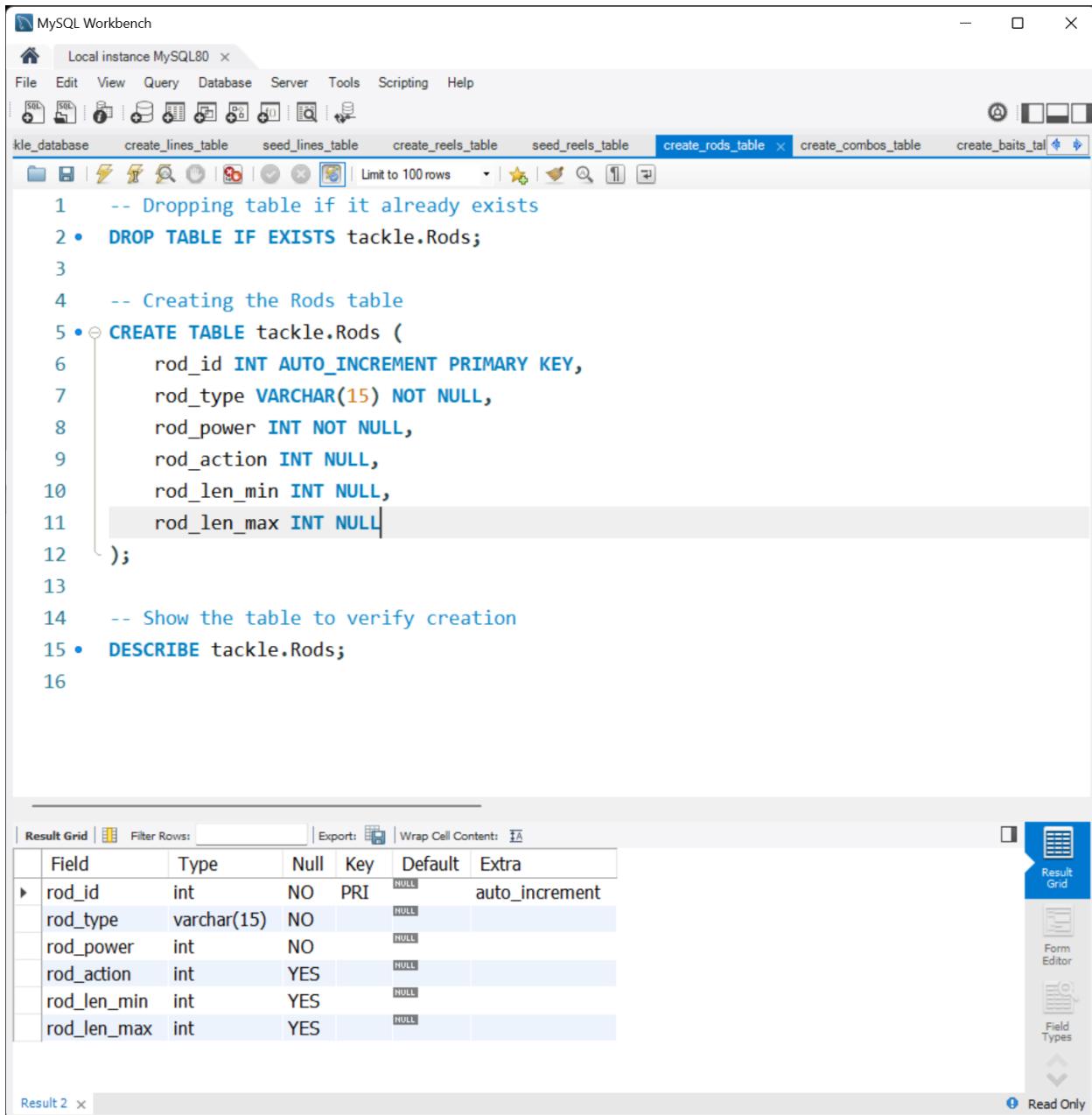
20      WHERE reel_type = 'cast' AND reel_gear_ratio = 6.4
21  );
22
23 • INSERT INTO tackle.Reels (reel_type, reel_gear_ratio)
24   SELECT 'spin', 5.8
25   WHERE NOT EXISTS (
26     SELECT 1 FROM tackle.Reels
27     WHERE reel_type = 'spin' AND reel_gear_ratio = 5.8
28  );
29
30 • INSERT INTO tackle.Reels (reel_type, reel_gear_ratio)
31   SELECT 'spin', 5.2
32   WHERE NOT EXISTS (
33     SELECT 1 FROM tackle.Reels
34     WHERE reel_type = 'spin' AND reel_gear_ratio = 5.2
35  );
36
37 -- Show the data to verify insertion
38 • SELECT * FROM tackle.Reels;
39

```

**Result Grid:**

reel_id	reel_type	reel_gear_ratio
1	cast	7.1
2	cast	5.4
3	cast	6.4
4	spin	5.8
5	spin	5.2
*	HULL	HULL

*Note.* Code at [https://github.com/ciiprof0/tackle/blob/main/db/seeds/seed\\_reels\\_table.sql](https://github.com/ciiprof0/tackle/blob/main/db/seeds/seed_reels_table.sql)  
 Seed data at <https://github.com/ciiprof0/tackle/raw/main/data/raw/data.xlsx>

**Figure 9***Create Rods Table*


The screenshot shows the MySQL Workbench interface with the following details:

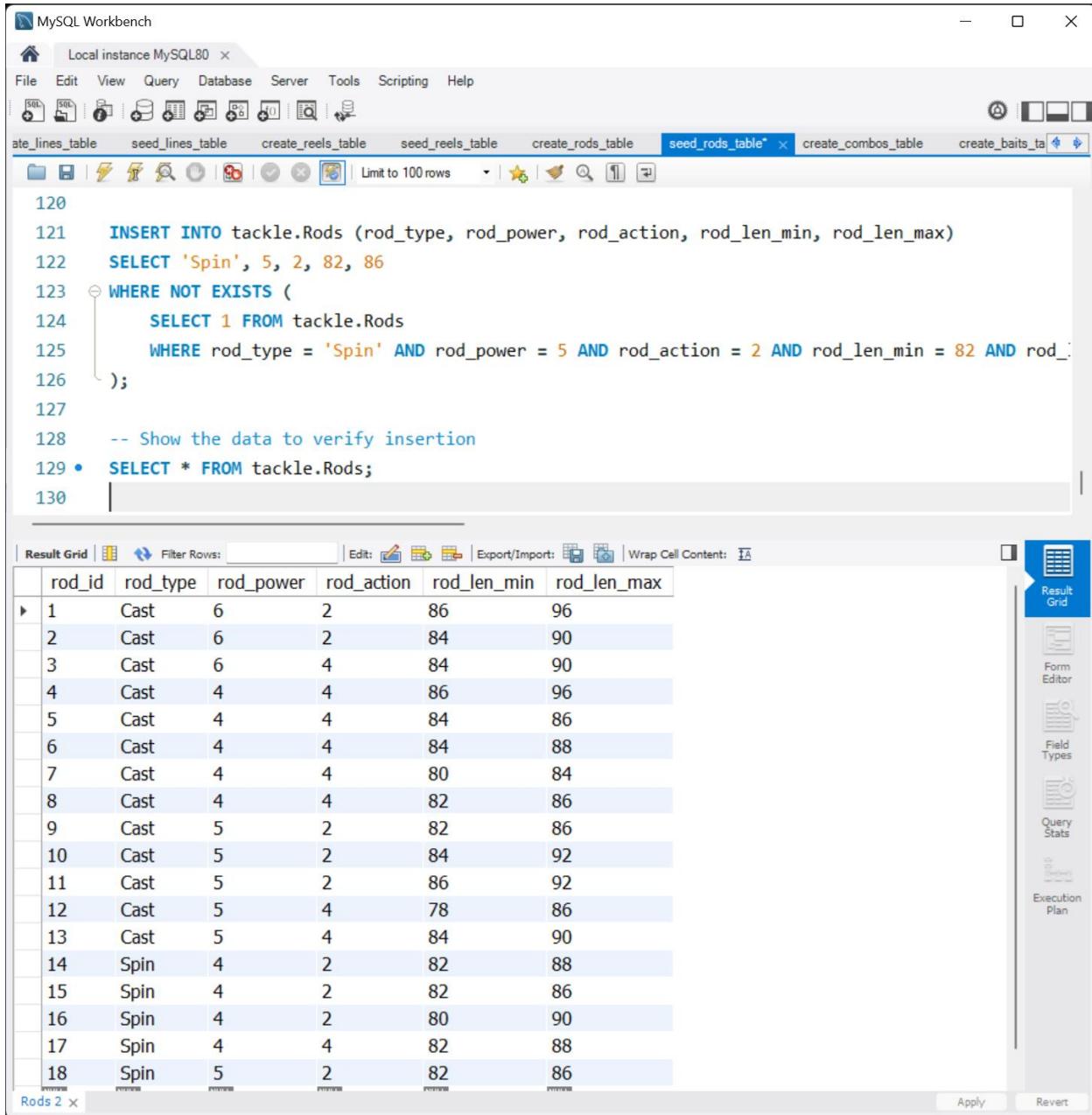
- Top Bar:** MySQL Workbench, Local instance MySQL80, File, Edit, View, Query, Database, Server, Tools, Scripting, Help.
- Toolbar:** SQL, DDL, Scripts, Data, Structure, Reports, Tasks, Utilities, Help.
- Query Editor:** A script window titled "create\_rod\_table" containing the following SQL code:
 

```

1 -- Dropping table if it already exists
2 • DROP TABLE IF EXISTS tackle.Rods;
3
4 -- Creating the Rods table
5 • CREATE TABLE tackle.Rods (
6     rod_id INT AUTO_INCREMENT PRIMARY KEY,
7     rod_type VARCHAR(15) NOT NULL,
8     rod_power INT NOT NULL,
9     rod_action INT NULL,
10    rod_len_min INT NULL,
11    rod_len_max INT NULL
12 );
13
14 -- Show the table to verify creation
15 • DESCRIBE tackle.Rods;
16
      
```
- Result Grid:** A table showing the structure of the Rods table with the following data:
 

Field	Type	Null	Key	Default	Extra
rod_id	int	NO	PRI	NULL	auto_increment
rod_type	varchar(15)	NO		NULL	
rod_power	int	NO		NULL	
rod_action	int	YES		NULL	
rod_len_min	int	YES		NULL	
rod_len_max	int	YES		NULL	
- Status Bar:** Result 2, Read Only.

*Note. Code at*[https://github.com/cioprof0/tackle/blob/main/db/schema/tables/create\\_rod\\_table.sql](https://github.com/cioprof0/tackle/blob/main/db/schema/tables/create_rod_table.sql)

**Figure 10***Seed Rods Table*


The screenshot shows the MySQL Workbench interface with a query editor and a result grid.

**Query Editor:**

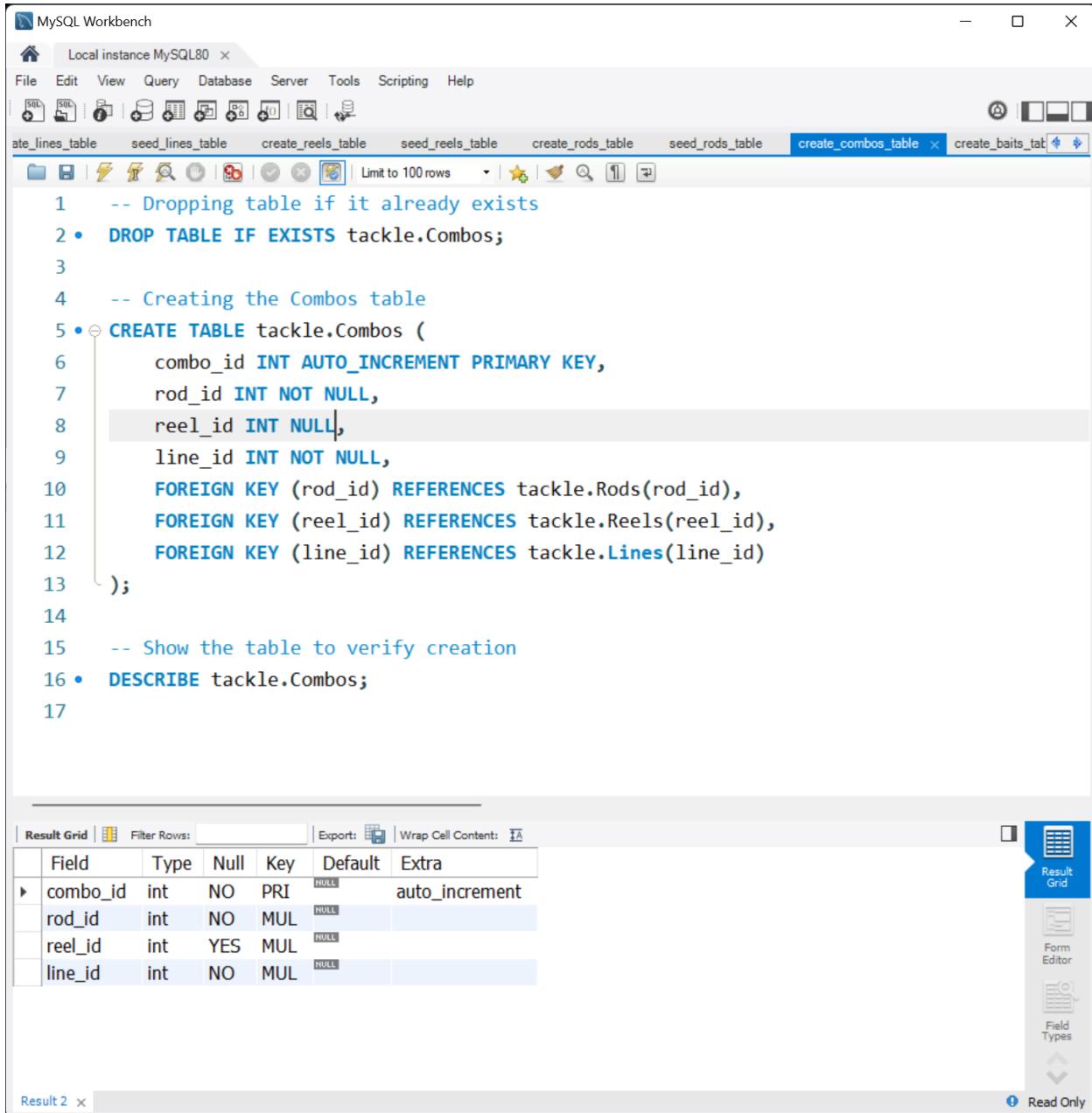
```

120
121     INSERT INTO tackle.Rods (rod_type, rod_power, rod_action, rod_len_min, rod_len_max)
122     SELECT 'Spin', 5, 2, 82, 86
123     WHERE NOT EXISTS (
124         SELECT 1 FROM tackle.Rods
125         WHERE rod_type = 'Spin' AND rod_power = 5 AND rod_action = 2 AND rod_len_min = 82 AND rod_
126     );
127
128     -- Show the data to verify insertion
129 •     SELECT * FROM tackle.Rods;
130
    
```

**Result Grid:**

rod_id	rod_type	rod_power	rod_action	rod_len_min	rod_len_max
1	Cast	6	2	86	96
2	Cast	6	2	84	90
3	Cast	6	4	84	90
4	Cast	4	4	86	96
5	Cast	4	4	84	86
6	Cast	4	4	84	88
7	Cast	4	4	80	84
8	Cast	4	4	82	86
9	Cast	5	2	82	86
10	Cast	5	2	84	92
11	Cast	5	2	86	92
12	Cast	5	4	78	86
13	Cast	5	4	84	90
14	Spin	4	2	82	88
15	Spin	4	2	82	86
16	Spin	4	2	80	90
17	Spin	4	4	82	88
18	Spin	5	2	82	86

*Note.* Code at [https://github.com/ciiprof0/tackle/blob/main/db/seeds/seed\\_rods\\_table.sql](https://github.com/ciiprof0/tackle/blob/main/db/seeds/seed_rods_table.sql)  
 Seed data at <https://github.com/ciiprof0/tackle/raw/main/data/raw/data.xlsx>

**Figure 11***Create Combos Table*


The screenshot shows the MySQL Workbench interface with the following details:

- File Bar:** File, Edit, View, Query, Database, Server, Tools, Scripting, Help.
- Toolbar:** Includes icons for Home, SQL, DDL, Scripts, Database, Tables, Views, Functions, Triggers, Events, and Procedures.
- Tab Bar:** Lists several tabs including 'ate\_lines\_table', 'seed\_lines\_table', 'create\_reels\_table', 'seed\_reels\_table', 'create\_rod\_table', 'seed\_rod\_table', 'create\_combos\_table' (which is currently selected), and 'create\_baits\_table'.
- Query Editor:** Displays the SQL code for creating the 'Combos' table. The code is as follows:

```

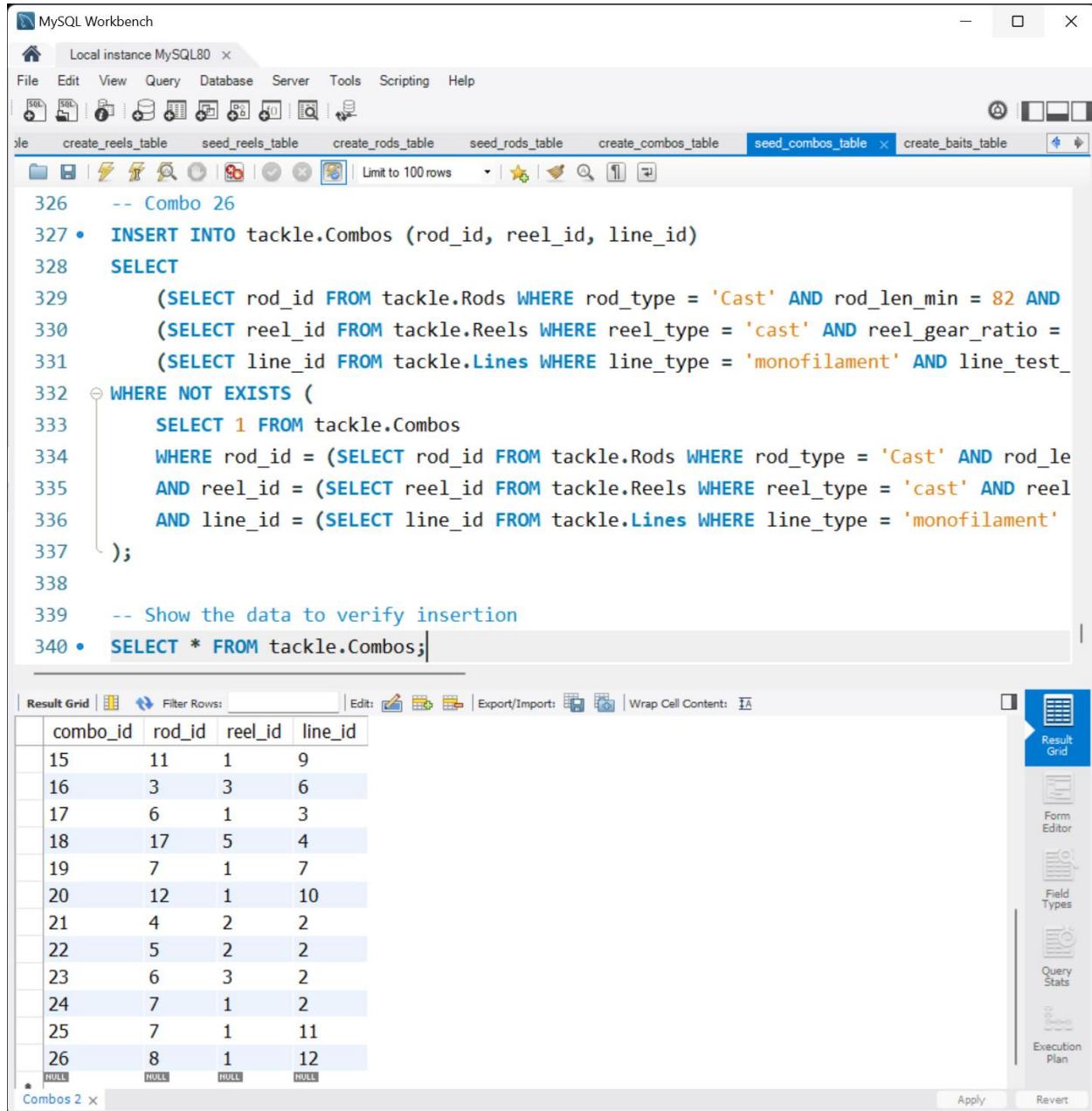
1 -- Dropping table if it already exists
2 • DROP TABLE IF EXISTS tackle.Combos;
3
4 -- Creating the Combos table
5 • CREATE TABLE tackle.Combos (
6     combo_id INT AUTO_INCREMENT PRIMARY KEY,
7     rod_id INT NOT NULL,
8     reel_id INT NULL,          -- This line is highlighted in gray
9     line_id INT NOT NULL,
10    FOREIGN KEY (rod_id) REFERENCES tackle.Rods(rod_id),
11    FOREIGN KEY (reel_id) REFERENCES tackle.Reels(reel_id),
12    FOREIGN KEY (line_id) REFERENCES tackle.Lines(line_id)
13 );
14
15 -- Show the table to verify creation
16 • DESCRIBE tackle.Combos;
17

```
- Result Grid:** Shows the structure of the 'Combos' table with the following columns:

Field	Type	Null	Key	Default	Extra
combo_id	int	NO	PRI	NULL	auto_increment
rod_id	int	NO	MUL	NULL	
reel_id	int	YES	MUL	NULL	
line_id	int	NO	MUL	NULL	
- Status Bar:** Shows 'Result 2' and 'Read Only'.

*Note.* Code at

[https://github.com/cioprof0/tackle/blob/main/db/schema/tables/create\\_combos\\_table.sql](https://github.com/cioprof0/tackle/blob/main/db/schema/tables/create_combos_table.sql)

**Figure 12***Seed Combos Table*


The screenshot shows the MySQL Workbench interface with a query editor and a result grid.

**Query Editor:**

```

326 -- Combo 26
327 • INSERT INTO tackle.Combos (rod_id, reel_id, line_id)
328 SELECT
329     (SELECT rod_id FROM tackle.Rods WHERE rod_type = 'Cast' AND rod_len_min = 82 AND
330      (SELECT reel_id FROM tackle.Reels WHERE reel_type = 'cast' AND reel_gear_ratio =
331      (SELECT line_id FROM tackle.Lines WHERE line_type = 'monofilament' AND line_test_
332      WHERE NOT EXISTS (
333          SELECT 1 FROM tackle.Combos
334          WHERE rod_id = (SELECT rod_id FROM tackle.Rods WHERE rod_type = 'Cast' AND rod_le
335          AND reel_id = (SELECT reel_id FROM tackle.Reels WHERE reel_type = 'cast' AND reel
336          AND line_id = (SELECT line_id FROM tackle.Lines WHERE line_type = 'monofilament'
337      );
338
339 -- Show the data to verify insertion
340 • SELECT * FROM tackle.Combos;

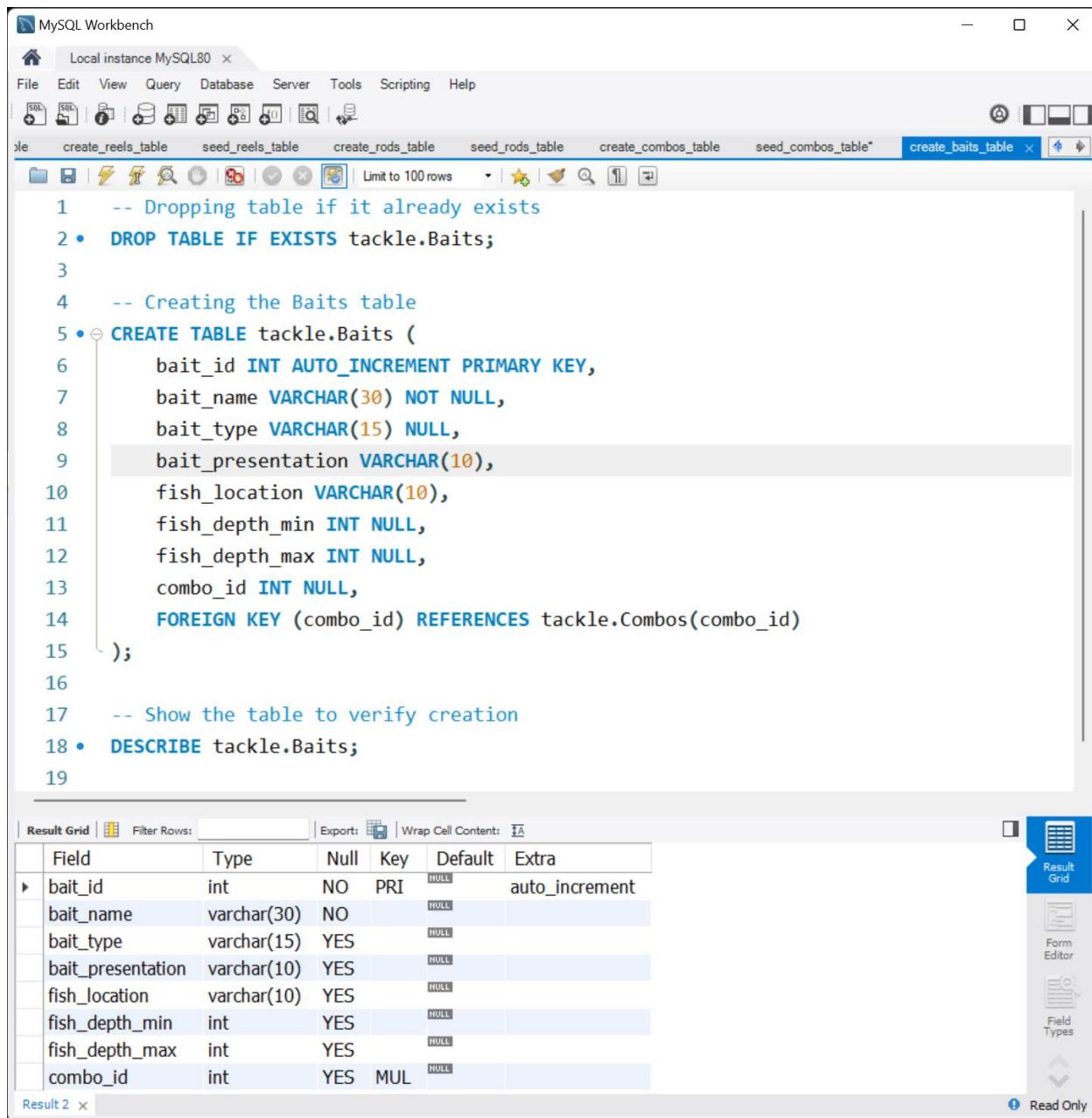
```

**Result Grid:**

combo_id	rod_id	reel_id	line_id
15	11	1	9
16	3	3	6
17	6	1	3
18	17	5	4
19	7	1	7
20	12	1	10
21	4	2	2
22	5	2	2
23	6	3	2
24	7	1	2
25	7	1	11
26	8	1	12
HULL	HULL	HULL	HULL

Combos 2 x

*Note.* Code at [https://github.com/ciiprof0/tackle/blob/main/db/seeds/seed\\_combos\\_table.sql](https://github.com/ciiprof0/tackle/blob/main/db/seeds/seed_combos_table.sql)  
 Seed data at <https://github.com/ciiprof0/tackle/raw/main/data/raw/data.xlsx>

**Figure 13***Create Baits Table*


The screenshot shows the MySQL Workbench interface with the following details:

- Title Bar:** MySQL Workbench - Local instance MySQL80
- Toolbar:** Standard MySQL Workbench toolbar with icons for Home, File, Edit, View, Query, Database, Server, Tools, Scripting, Help, and various database objects.
- Script Editor:** The main area contains an SQL script for creating the 'Baits' table. The code is as follows:

```

1 -- Dropping table if it already exists
2 • DROP TABLE IF EXISTS tackle.Baits;
3
4 -- Creating the Baits table
5 • CREATE TABLE tackle.Baits (
6     bait_id INT AUTO_INCREMENT PRIMARY KEY,
7     bait_name VARCHAR(30) NOT NULL,
8     bait_type VARCHAR(15) NULL,
9     bait_presentation VARCHAR(10),
10    fish_location VARCHAR(10),
11    fish_depth_min INT NULL,
12    fish_depth_max INT NULL,
13    combo_id INT NULL,
14    FOREIGN KEY (combo_id) REFERENCES tackle.Combos(combo_id)
15 );
16
17 -- Show the table to verify creation
18 • DESCRIBE tackle.Baits;
19

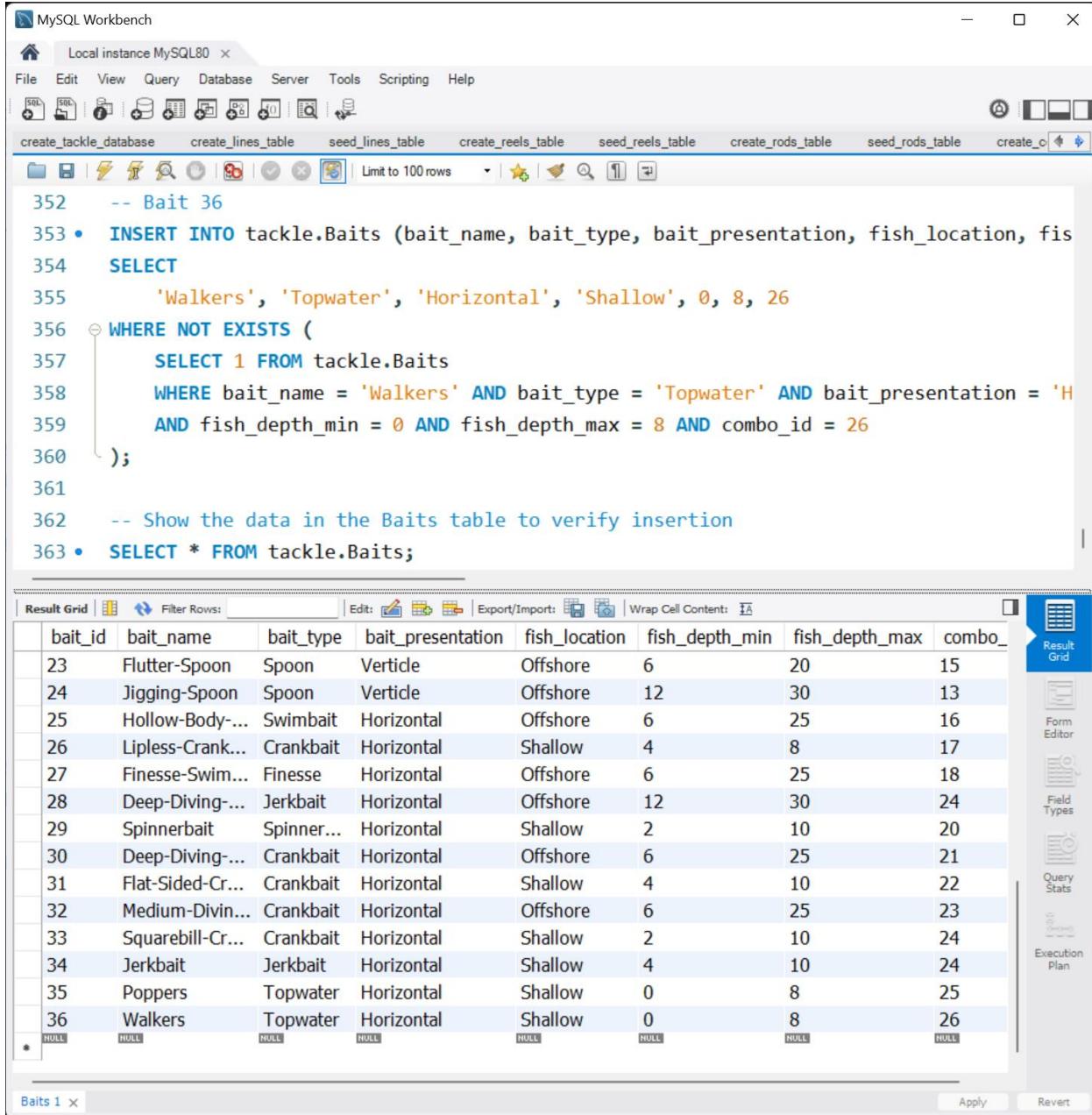
```

- Result Grid:** Below the script editor, a table displays the structure of the 'Baits' table:

Field	Type	Null	Key	Default	Extra
bait_id	int	NO	PRI	NULL	auto_increment
bait_name	varchar(30)	NO		NULL	
bait_type	varchar(15)	YES		NULL	
bait_presentation	varchar(10)	YES		NULL	
fish_location	varchar(10)	YES		NULL	
fish_depth_min	int	YES		NULL	
fish_depth_max	int	YES		NULL	
combo_id	int	YES	MUL	NULL	

- Status Bar:** Shows 'Result 2' and 'Read Only' status.

*Note. Code at*[https://github.com/cioprof0/tackle/blob/main/db/schema/tables/create\\_baits\\_table.sql](https://github.com/cioprof0/tackle/blob/main/db/schema/tables/create_baits_table.sql)

**Figure 14***Seed Baits Table*


The screenshot shows the MySQL Workbench interface with a query editor and a result grid.

**Query Editor:**

```

352 -- Bait 36
353 • INSERT INTO tackle.Baits (bait_name, bait_type, bait_presentation, fish_location, fis
354     SELECT
355         'Walkers', 'Topwater', 'Horizontal', 'Shallow', 0, 8, 26
356     WHERE NOT EXISTS (
357         SELECT 1 FROM tackle.Baits
358         WHERE bait_name = 'Walkers' AND bait_type = 'Topwater' AND bait_presentation = 'H
359             AND fish_depth_min = 0 AND fish_depth_max = 8 AND combo_id = 26
360     );
361
362 -- Show the data in the Baits table to verify insertion
363 • SELECT * FROM tackle.Baits;

```

**Result Grid:**

bait_id	bait_name	bait_type	bait_presentation	fish_location	fish_depth_min	fish_depth_max	combo_id
23	Flutter-Spoon	Spoon	Verticle	Offshore	6	20	15
24	Jigging-Spoon	Spoon	Verticle	Offshore	12	30	13
25	Hollow-Body....	Swimbait	Horizontal	Offshore	6	25	16
26	Lipless-Crank...	Crankbait	Horizontal	Shallow	4	8	17
27	Finesse-Swim...	Finesse	Horizontal	Offshore	6	25	18
28	Deep-Diving....	Jerkbait	Horizontal	Offshore	12	30	24
29	Spinnerbait	Spinner...	Horizontal	Shallow	2	10	20
30	Deep-Diving....	Crankbait	Horizontal	Offshore	6	25	21
31	Flat-Sided-Cr...	Crankbait	Horizontal	Shallow	4	10	22
32	Medium-Divin...	Crankbait	Horizontal	Offshore	6	25	23
33	Squarebill-Cr...	Crankbait	Horizontal	Shallow	2	10	24
34	Jerkbait	Jerkbait	Horizontal	Shallow	4	10	24
35	Poppers	Topwater	Horizontal	Shallow	0	8	25
36	Walkers	Topwater	Horizontal	Shallow	0	8	26

*Note.* Code at [https://github.com/ciiprof0/tackle/blob/main/db/seeds/seed\\_baits\\_table.sql](https://github.com/ciiprof0/tackle/blob/main/db/seeds/seed_baits_table.sql)  
 Seed data at <https://github.com/ciiprof0/tackle/raw/main/data/raw/data.xlsx>

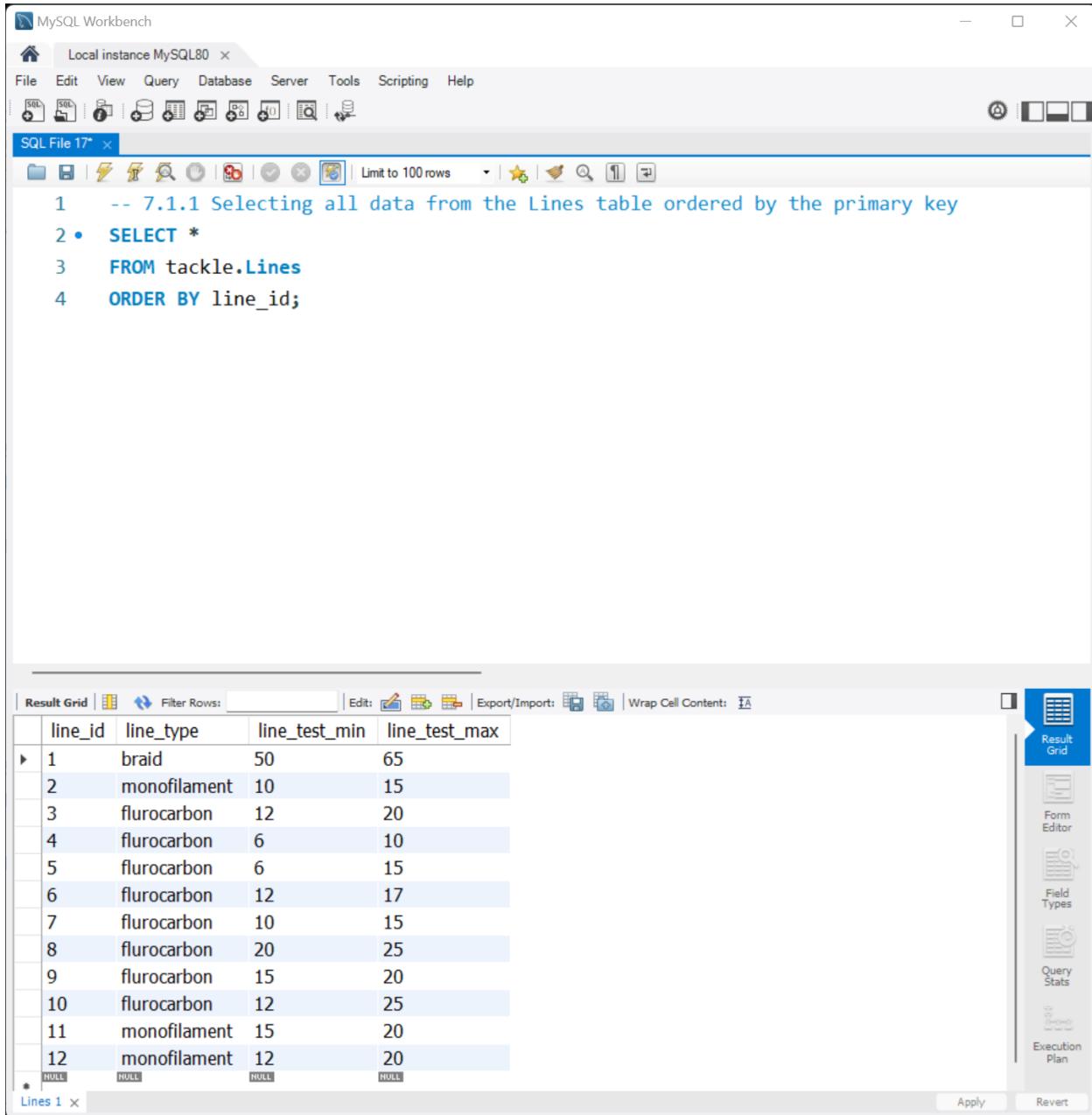
## Queries

7. Create a variety of SQL queries to retrieve data from one or many tables:

*1. Retrieve the data from each table by using the SELECT \* statement and order by PK column(s). Show the output. Make sure you show the print screen of the complete set of rows and columns. The rows must be ordered by PK column(s).*

**Figure 15**

*SELECT \* FROM Lines Table*



The screenshot shows the MySQL Workbench interface. The title bar says "MySQL Workbench" and "Local instance MySQL80". The menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. Below the menu is a toolbar with various icons. The main area has a tab labeled "SQL File 17\*" which contains the following SQL code:

```

1 -- 7.1.1 Selecting all data from the Lines table ordered by the primary key
2 • SELECT *
3 FROM tackle.Lines
4 ORDER BY line_id;

```

Below the code, the "Result Grid" pane displays the data from the "Lines" table. The columns are "line\_id", "line\_type", "line\_test\_min", and "line\_test\_max". The data is as follows:

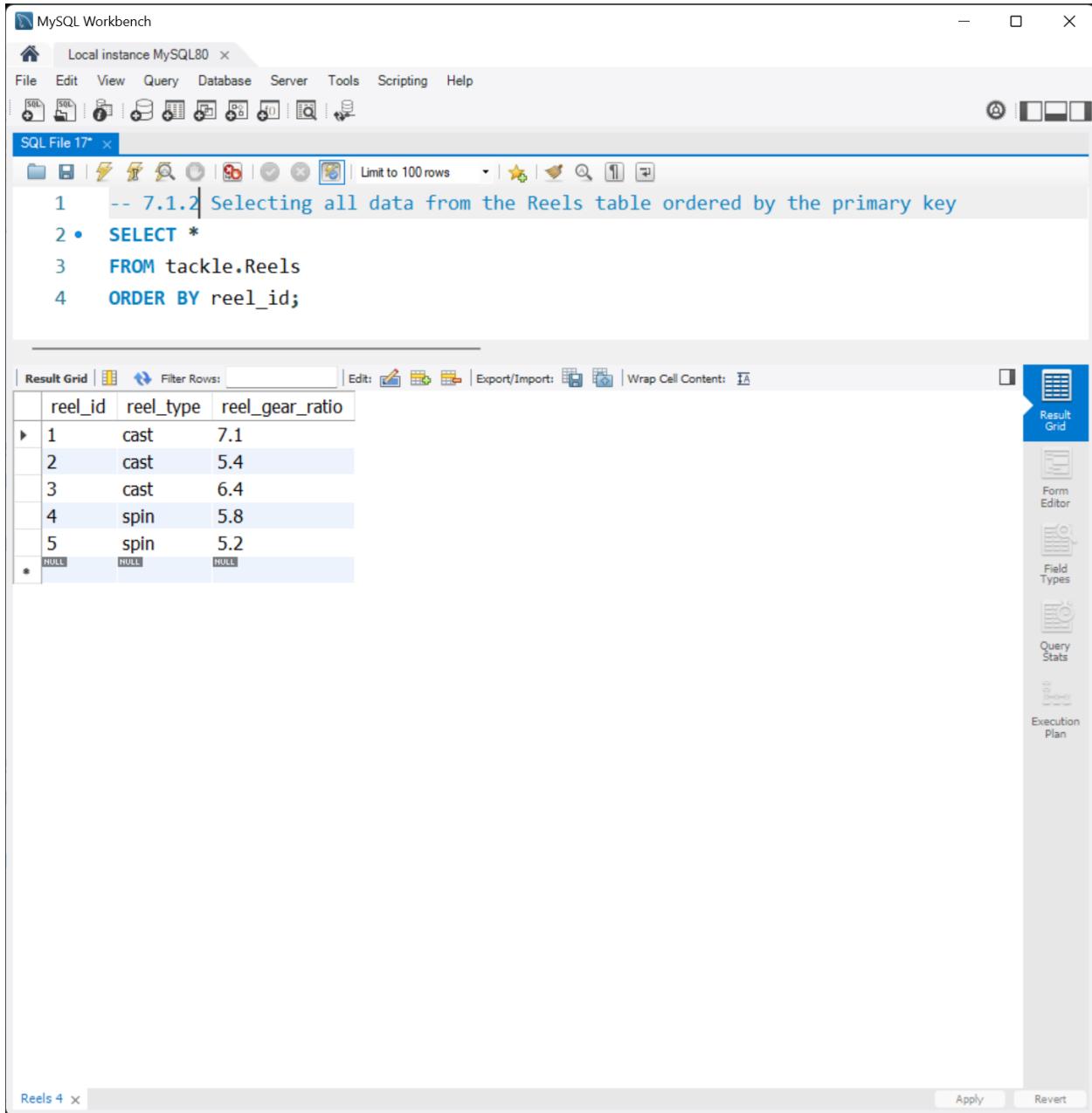
line_id	line_type	line_test_min	line_test_max
1	braid	50	65
2	monofilament	10	15
3	fluorocarbon	12	20
4	fluorocarbon	6	10
5	fluorocarbon	6	15
6	fluorocarbon	12	17
7	fluorocarbon	10	15
8	fluorocarbon	20	25
9	fluorocarbon	15	20
10	fluorocarbon	12	25
11	monofilament	15	20
12	monofilament	12	20
HULL	HULL	HULL	HULL

The "Result Grid" tab is selected in the bottom right corner of the results pane. Other tabs include "Form Editor", "Field Types", "Query Stats", and "Execution Plan".

*Note.* Code at <https://github.com/cioprof0/tackle/blob/main/db/schema/queries/q71.sql>.  
Data at [https://github.com/cioprof0/tackle/blob/main/data/processed/7\\_1\\_1.csv](https://github.com/cioprof0/tackle/blob/main/data/processed/7_1_1.csv)

**Figure 16**

*SELECT \* FROM Reels Table*



The screenshot shows the MySQL Workbench interface. The top menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. Below the menu is a toolbar with various icons. The main area has a tab labeled "SQL File 17" containing the following SQL code:

```

1 -- 7.1.2 Selecting all data from the Reels table ordered by the primary key
2 • SELECT *
3   FROM tackle.Reels
4   ORDER BY reel_id;

```

Below the code is a "Result Grid" table with the following data:

reel_id	reel_type	reel_gear_ratio
1	cast	7.1
2	cast	5.4
3	cast	6.4
4	spin	5.8
5	spin	5.2
*	HULL	HULL

The right side of the interface features a sidebar with several tabs: Result Grid (selected), Form Editor, Field Types, Query Stats, and Execution Plan.

*Note.* Code at <https://github.com/ciiprof0/tackle/blob/main/db/schema/queries/q71.sql>.  
Data at [https://github.com/ciiprof0/tackle/blob/main/data/processed/7\\_1\\_2.csv](https://github.com/ciiprof0/tackle/blob/main/data/processed/7_1_2.csv)

**Figure 17**

*SELECT \* FROM Rods Table*

The screenshot shows the MySQL Workbench interface with a query editor and a results grid.

**Query Editor:**

```

1 -- 7.1.3 Selecting all data from the Rods table ordered by the primary key
2 • SELECT *
3   FROM tackle.Rods
4   ORDER BY rod_id;
    
```

**Result Grid:**

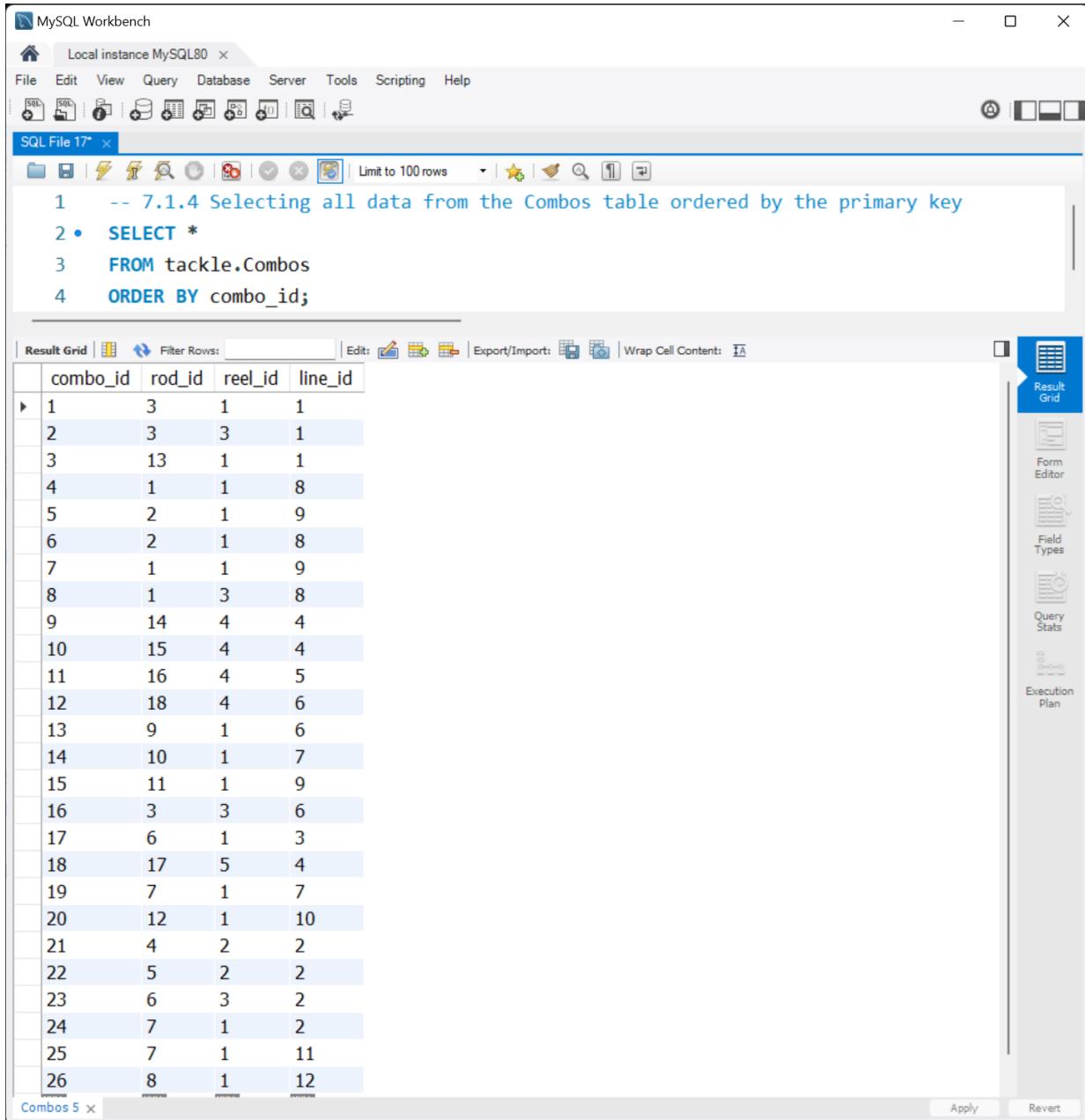
rod_id	rod_type	rod_power	rod_action	rod_len_min	rod_len_max
1	Cast	6	2	86	96
2	Cast	6	2	84	90
3	Cast	6	4	84	90
4	Cast	4	4	86	96
5	Cast	4	4	84	86
6	Cast	4	4	84	88
7	Cast	4	4	80	84
8	Cast	4	4	82	86
9	Cast	5	2	82	86
10	Cast	5	2	84	92
11	Cast	5	2	86	92
12	Cast	5	4	78	86
13	Cast	5	4	84	90
14	Spin	4	2	82	88
15	Spin	4	2	82	86
16	Spin	4	2	80	90
17	Spin	4	4	82	88
18	Spin	5	2	82	86
*	HULL	HULL	HULL	HULL	HULL

The results show 18 rows of data from the Rods table, ordered by rod\_id. The columns are rod\_id, rod\_type, rod\_power, rod\_action, rod\_len\_min, and rod\_len\_max. The data includes various rod types like Cast and Spin with different power levels and action counts, and length ranges from 78 to 96.

*Note.* Code at <https://github.com/cioprof0/tackle/blob/main/db/schema/queries/q71.sql>.  
Data at [https://github.com/cioprof0/tackle/blob/main/data/processed/7\\_1\\_3.csv](https://github.com/cioprof0/tackle/blob/main/data/processed/7_1_3.csv)

**Figure 18**

*SELECT \* FROM Combo Table*



The screenshot shows the MySQL Workbench interface with a query editor window. The title bar says "MySQL Workbench" and "Local instance MySQL80". The menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. Below the menu is a toolbar with various icons. The query tab is titled "SQL File 17" and contains the following SQL code:

```

1 -- 7.1.4 Selecting all data from the Combos table ordered by the primary key
2 • SELECT *
3   FROM tackle.Combos
4 ORDER BY combo_id;

```

The results are displayed in a "Result Grid" table. The columns are labeled "combo\_id", "rod\_id", "reel\_id", and "line\_id". The data consists of 26 rows of integer values. The table has a header row and 26 data rows. The "Result Grid" tab is selected in the sidebar on the right.

	combo_id	rod_id	reel_id	line_id
1	3	1	1	
2	3	3	1	
3	13	1	1	
4	1	1	8	
5	2	1	9	
6	2	1	8	
7	1	1	9	
8	1	3	8	
9	14	4	4	
10	15	4	4	
11	16	4	5	
12	18	4	6	
13	9	1	6	
14	10	1	7	
15	11	1	9	
16	3	3	6	
17	6	1	3	
18	17	5	4	
19	7	1	7	
20	12	1	10	
21	4	2	2	
22	5	2	2	
23	6	3	2	
24	7	1	2	
25	7	1	11	
26	8	1	12	

*Note.* Code at <https://github.com/cioprof0/tackle/blob/main/db/schema/queries/q71.sql>.  
Data at [https://github.com/cioprof0/tackle/blob/main/data/processed/7\\_1\\_4.csv](https://github.com/cioprof0/tackle/blob/main/data/processed/7_1_4.csv)

**Figure 19**

*SELECT \* FROM Baits Table*

The screenshot shows the MySQL Workbench interface with a SQL editor window and a results grid window.

**SQL Editor:**

```
2 • SELECT *
3   FROM tackle.Baits
4 ORDER BY bait_id;
```

**Results Grid:**

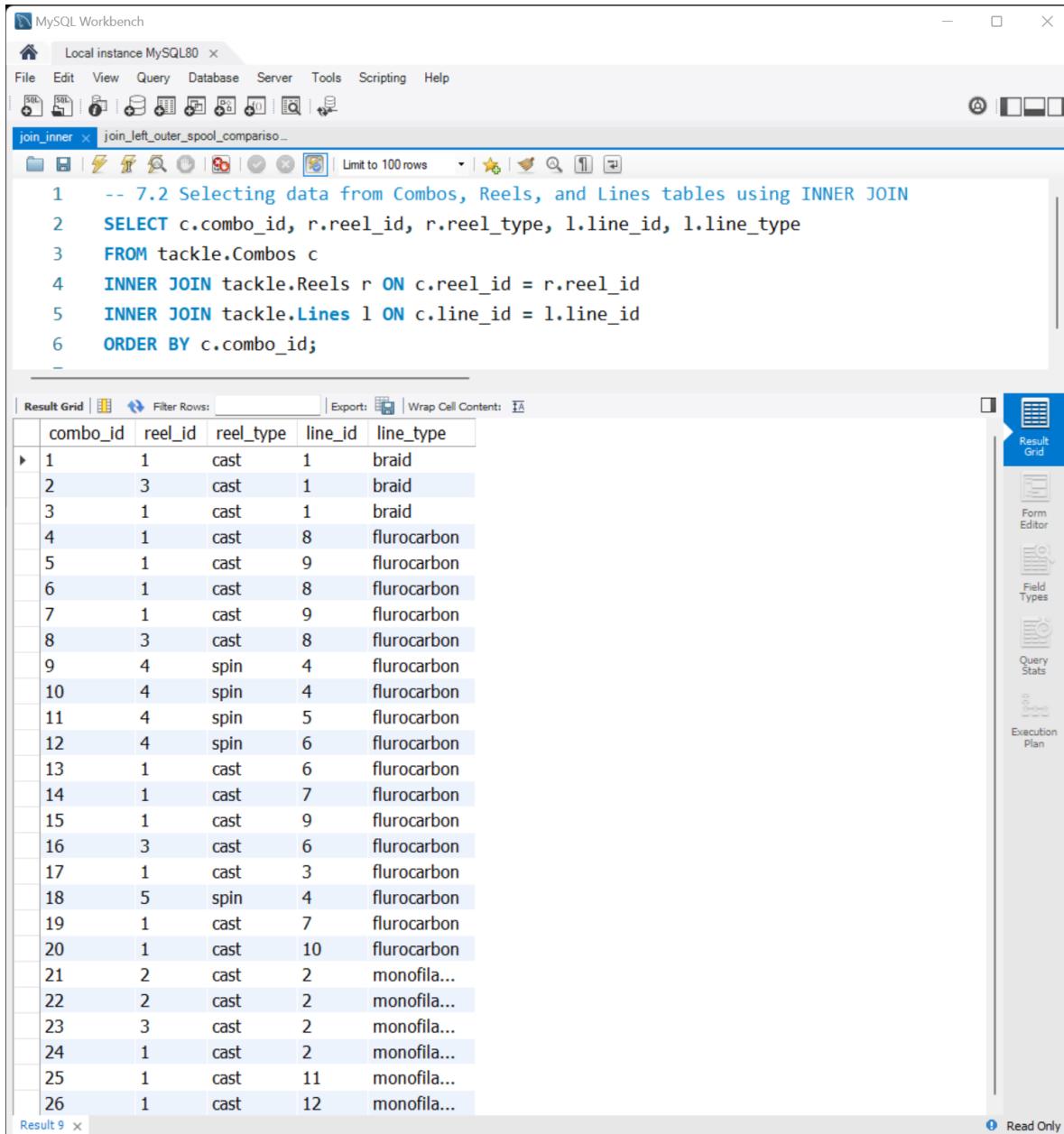
bait_id	bait_name	bait_type	bait_presentation	fish_location	fish_depth_min	fish_depth_max	combo_id
1	Buzzbait	Buzz	Horizontal	Shallow	4	10	1
2	Ploppers	Topwater	Horizontal	Shallow	0	8	2
3	Topwater-Frog	Topwater	Horizontal	Shallow	0	12	3
4	Flipping-Jig	Jig	Verticle	Shallow	2	10	4
5	Football-Jig	Jig	Verticle	Offshore	6	25	5
6	Swim-Jig	Jig	Horizontal	Shallow	4	10	6
7	Carolina-Rig	Rig	Horizontal	Offshore	6	20	4
8	Creature-Bait	Rig	Verticle	Shallow	2	10	5
9	Swing-Head	Rig	Verticle	Offshore	6	20	5
10	Umbrella-Rig	Rig	Horizontal	Shallow	2	10	8
11	Worm-10in	Rig	Verticle	Offshore	6	25	5
12	Drop-Shot	Finesse	Verticle	Offshore	6	25	9
13	Ned-Rig	Finesse	Verticle	Shallow	1	10	10
14	Neko-Rig	Finesse	Verticle	Shallow	1	10	9
15	Shakey-Head	Finesse	Verticle	Shallow	4	10	10
16	Tube	Finesse	Verticle	Offshore	4	12	11
17	Wacky-Rig	Finesse	Verticle	Shallow	1	10	10
18	Soft-Jerkbait	Jerkbait	Horizontal	Shallow	1	10	9
19	Stick-Worm	Finesse	Verticle	Shallow	2	10	12
20	Finesse-Jig	Finesse	Verticle	Shallow	4	10	13
21	Hair-Jig	Jig	Verticle	Offshore	6	25	14
22	Worm-7in	Rig	Verticle	Shallow	1	10	13
23	Flutter-Spoon	Spoon	Verticle	Offshore	6	20	15
24	Jigging-Spoon	Spoon	Verticle	Offshore	12	30	13
25	Hollow-Body...	Swimbait	Horizontal	Offshore	6	25	16
26	Lipless-Crank...	Crankbait	Horizontal	Shallow	4	8	17
27	Finesse-Swim...	Finesse	Horizontal	Offshore	6	25	18
28	Deep-Diving...	Jerkbait	Horizontal	Offshore	12	30	24
29	Spinnerbait	Spinner...	Horizontal	Shallow	2	10	20
30	Deep-Diving...	Crankbait	Horizontal	Offshore	6	25	21
31	Flat-Sided-Cr...	Crankbait	Horizontal	Shallow	4	10	22
32	Medium-Divin...	Crankbait	Horizontal	Offshore	6	25	23
33	Squarebill-Cr...	Crankbait	Horizontal	Shallow	2	10	24
34	Jerkbait	Jerkbait	Horizontal	Shallow	4	10	24
35	Poppers	Topwater	Horizontal	Shallow	0	8	25
36	Walkers	Topwater	Horizontal	Shallow	0	8	26

*Note.* Code at <https://github.com/ciiprof0/tackle/blob/main/db/schema/queries/q71.sql>.  
Data at [https://github.com/ciiprof0/tackle/blob/main/data/processed/7\\_1\\_5.csv](https://github.com/ciiprof0/tackle/blob/main/data/processed/7_1_5.csv)

**2. Write an SQL involving the junction table and two other related tables. You must use the INNER JOIN to connect with all three tables. The database that you created must be included in your SQL queries.**

**Figure 20**

*INNER JOIN Connecting Three Tables*



The screenshot shows the MySQL Workbench interface with a query editor and a result grid. The query editor contains the following SQL code:

```

1 -- 7.2 Selecting data from Combos, Reels, and Lines tables using INNER JOIN
2 SELECT c.combo_id, r.reel_id, r.reel_type, l.line_id, l.line_type
3 FROM tackle.Combos c
4 INNER JOIN tackle.Reels r ON c.reel_id = r.reel_id
5 INNER JOIN tackle.Lines l ON c.line_id = l.line_id
6 ORDER BY c.combo_id;

```

The result grid displays the following data:

combo_id	reel_id	reel_type	line_id	line_type
1	1	cast	1	braid
2	3	cast	1	braid
3	1	cast	1	braid
4	1	cast	8	flurocarbon
5	1	cast	9	flurocarbon
6	1	cast	8	flurocarbon
7	1	cast	9	flurocarbon
8	3	cast	8	flurocarbon
9	4	spin	4	flurocarbon
10	4	spin	4	flurocarbon
11	4	spin	5	flurocarbon
12	4	spin	6	flurocarbon
13	1	cast	6	flurocarbon
14	1	cast	7	flurocarbon
15	1	cast	9	flurocarbon
16	3	cast	6	flurocarbon
17	1	cast	3	flurocarbon
18	5	spin	4	flurocarbon
19	1	cast	7	flurocarbon
20	1	cast	10	flurocarbon
21	2	cast	2	monofil...
22	2	cast	2	monofil...
23	3	cast	2	monofil...
24	1	cast	2	monofil...
25	1	cast	11	monofil...
26	1	cast	12	monofil...

*Note.* Code at <https://github.com/cioprof0/tackle/blob/main/db/schema/queries/q72.sql>.  
Data at [https://github.com/cioprof0/tackle/blob/main/data/processed/7\\_2.csv](https://github.com/cioprof0/tackle/blob/main/data/processed/7_2.csv)

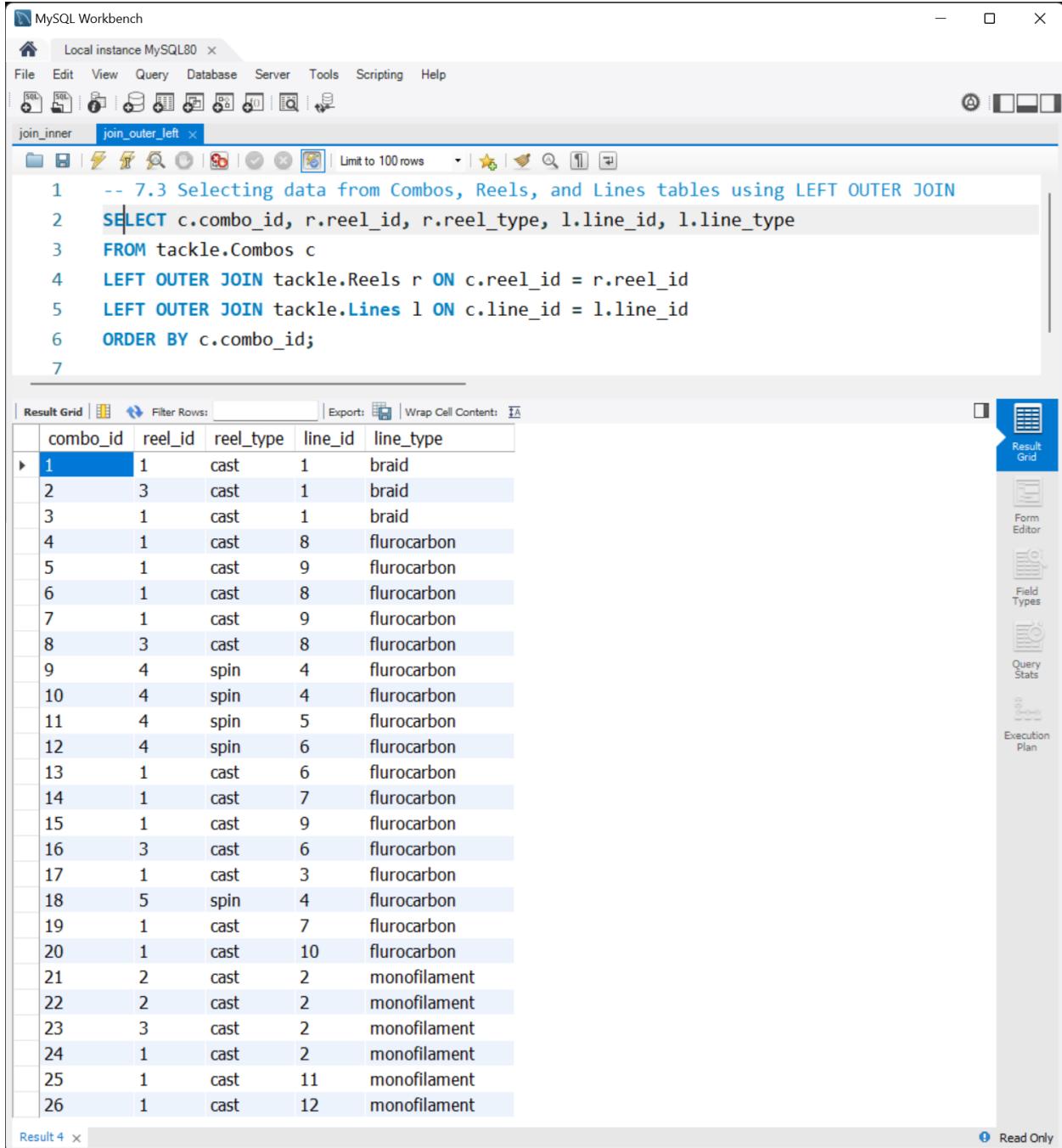
**3. Write an SQL by including two or more tables and using the LEFT OUTER JOIN. Show the results and sort the results by key field(s). Interpret the results compared to what an INNER JOIN does.**

The results are the same because the `tackle` dataset is "complete" regarding relationships between the Combos, Reels, and Lines tables. Since every `combo_id` has matching `reel_id` and `line_id`, both types of joins return identical results.

To demonstrate the difference between INNER JOIN and LEFT OUTER JOIN, we could introduce a scenario where a `combo_id` references a `reel_id` or `line_id` that does not exist in the corresponding tables. This would allow us to see how LEFT OUTER JOIN behaves differently by including rows with NULL values.

**Figure 21**

*LEFT OUTER JOIN Connecting Three Tables*



The screenshot shows the MySQL Workbench interface with a query editor and a results grid.

**Query Editor:**

```

1 -- 7.3 Selecting data from Combos, Reels, and Lines tables using LEFT OUTER JOIN
2 SELECT c.combo_id, r.reel_id, r.reel_type, l.line_id, l.line_type
3 FROM tackle.Combos c
4 LEFT OUTER JOIN tackle.Reels r ON c.reel_id = r.reel_id
5 LEFT OUTER JOIN tackle.Lines l ON c.line_id = l.line_id
6 ORDER BY c.combo_id;
7
  
```

**Result Grid:**

combo_id	reel_id	reel_type	line_id	line_type
1	1	cast	1	braid
2	3	cast	1	braid
3	1	cast	1	braid
4	1	cast	8	flurocarbon
5	1	cast	9	flurocarbon
6	1	cast	8	flurocarbon
7	1	cast	9	flurocarbon
8	3	cast	8	flurocarbon
9	4	spin	4	flurocarbon
10	4	spin	4	flurocarbon
11	4	spin	5	flurocarbon
12	4	spin	6	flurocarbon
13	1	cast	6	flurocarbon
14	1	cast	7	flurocarbon
15	1	cast	9	flurocarbon
16	3	cast	6	flurocarbon
17	1	cast	3	flurocarbon
18	5	spin	4	flurocarbon
19	1	cast	7	flurocarbon
20	1	cast	10	flurocarbon
21	2	cast	2	monofilament
22	2	cast	2	monofilament
23	3	cast	2	monofilament
24	1	cast	2	monofilament
25	1	cast	11	monofilament
26	1	cast	12	monofilament

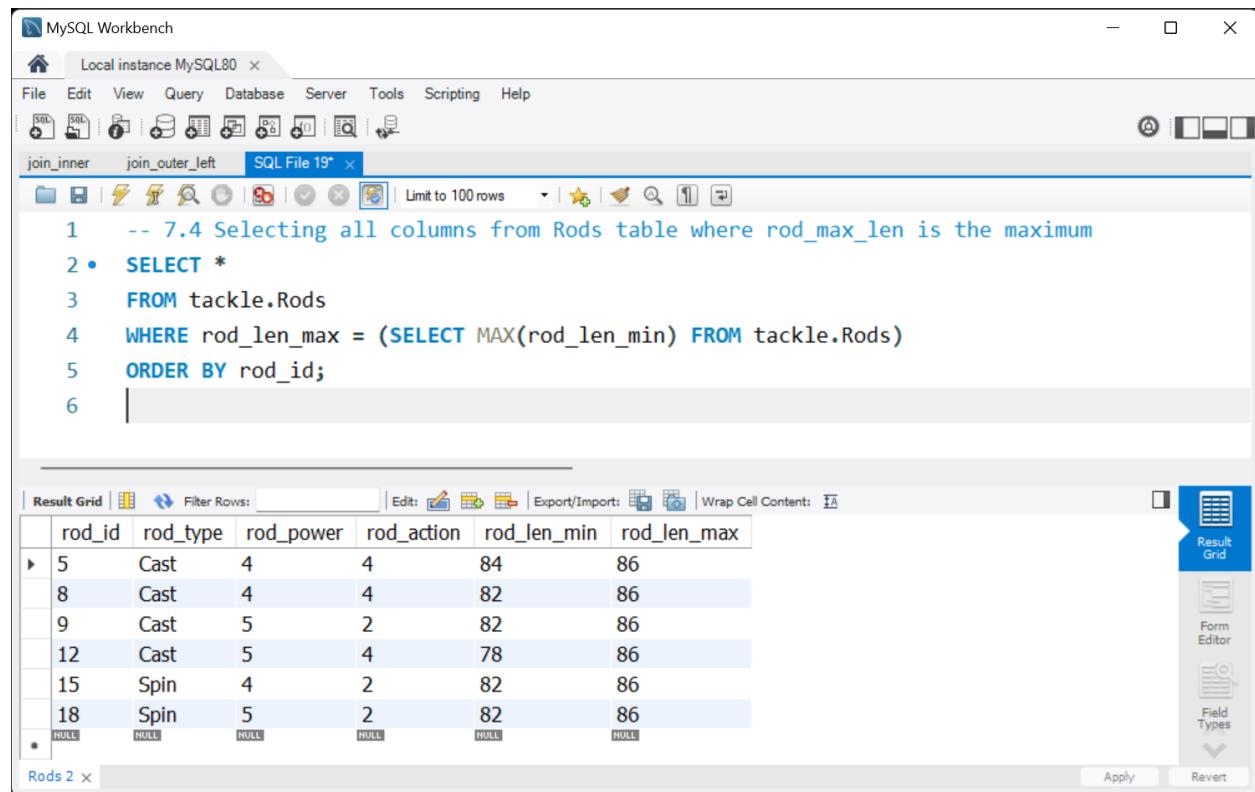
*Note.* Code at <https://github.com/ciiprof0/tackle/blob/main/db/schema/queries/q73.sql>.  
Data at [https://github.com/ciiprof0/tackle/blob/main/data/processed/7\\_3.csv](https://github.com/ciiprof0/tackle/blob/main/data/processed/7_3.csv)

**4. Write a single-row subquery. Show the results and sort the results by key field(s). Interpret the output.**

The results show there are six rods whose maximum length (`rod_len_max`) equals the longest minimum length (`rod_len_min`) rod(s). The variability in `rod_len_min` suggests differences in the rods' minimum usable lengths, but all reach the same maximum extension, highlighting their potential for extended use.

**Figure 22**

*Single-Row Subquery*



The screenshot shows the MySQL Workbench interface. The top window is titled "MySQL Workbench" and has a tab bar with "join\_inner" and "join\_outer\_left" selected. Below the tabs is a toolbar with various icons. The main area contains a SQL editor with the following code:

```

1 -- 7.4 Selecting all columns from Rods table where rod_max_len is the maximum
2 • SELECT *
3   FROM tackle.Rods
4 WHERE rod_len_max = (SELECT MAX(rod_len_min) FROM tackle.Rods)
5 ORDER BY rod_id;
6

```

Below the SQL editor is a "Result Grid" window. It displays a table with the following data:

rod_id	rod_type	rod_power	rod_action	rod_len_min	rod_len_max
5	Cast	4	4	84	86
8	Cast	4	4	82	86
9	Cast	5	2	82	86
12	Cast	5	4	78	86
15	Spin	4	2	82	86
18	Spin	5	2	82	86
*	NULL	NULL	NULL	NULL	NULL

The "Result Grid" tab is selected in the bottom-left corner of the Result Grid window. On the right side of the Result Grid window, there are three buttons: "Result Grid" (selected), "Form Editor", and "Field Types".

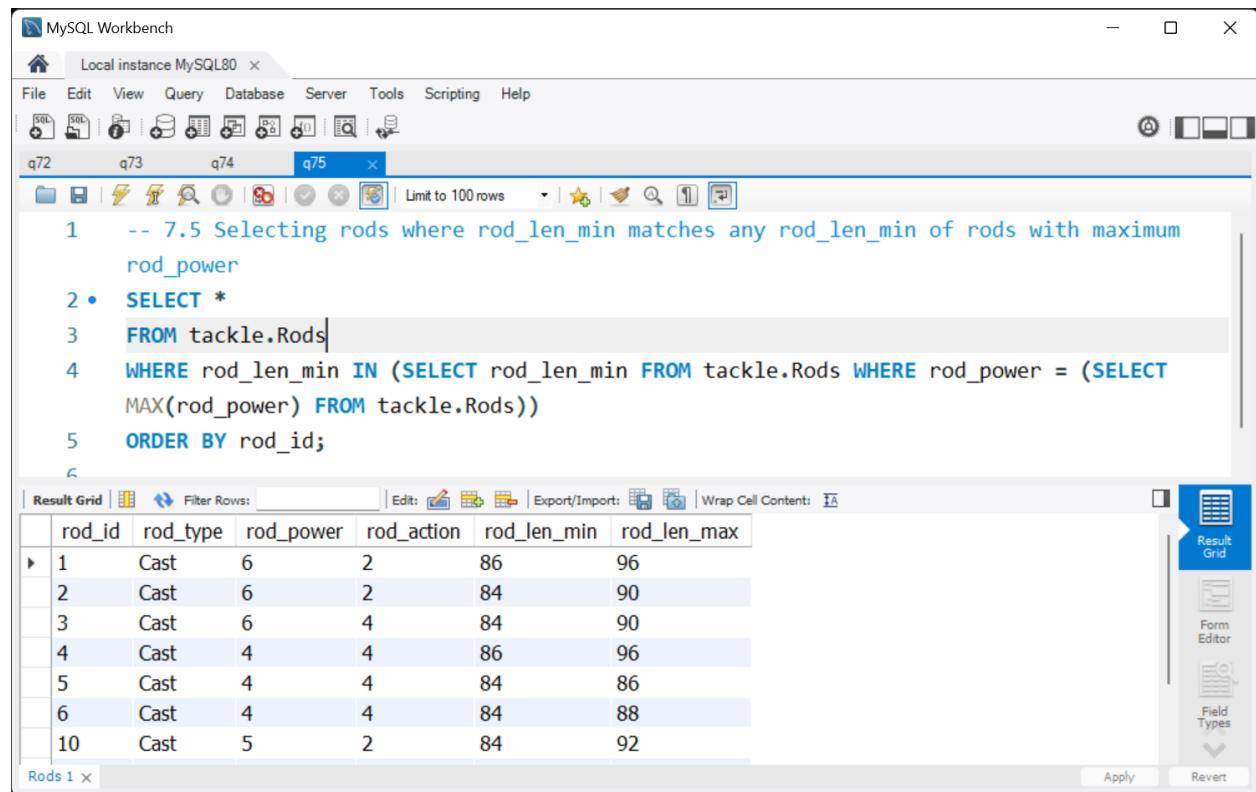
*Note.* Code at <https://github.com/ciiprof0/tackle/blob/main/db/schema/queries/q74.sql>.  
Data at [https://github.com/ciiprof0/tackle/blob/main/data/processed/7\\_4.csv](https://github.com/ciiprof0/tackle/blob/main/data/processed/7_4.csv)

**5. Write a multiple-row subquery. Show the results and sort the results by key field(s). Interpret the output.**

The results show rods with a `rod_len_min` matching the minimum lengths of rods with the highest power (`rod_power = 6`). The query highlights that rods with varying power levels ('4', '5', and '6') share similar minimum lengths ('84' or '86'), indicating a common design choice in rod lengths across different power categories. All rods listed are of the "Cast" type, suggesting uniformity in design within this category.

**Figure 23**

*Multiple-Row Subquery*



The screenshot shows the MySQL Workbench interface. The title bar says "MySQL Workbench" and "Local instance MySQL80". The menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. Below the menu is a toolbar with various icons. A tab bar at the top has tabs q72, q73, q74, and q75, with q75 currently selected. The main area contains a SQL editor with the following code:

```

1 -- 7.5 Selecting rods where rod_len_min matches any rod_len_min of rods with maximum
2 • SELECT *
3   FROM tackle.Rods
4 WHERE rod_len_min IN (SELECT rod_len_min FROM tackle.Rods WHERE rod_power = (SELECT
      MAX(rod_power) FROM tackle.Rods))
5 ORDER BY rod_id;
6

```

Below the SQL editor is a "Result Grid" table with the following data:

rod_id	rod_type	rod_power	rod_action	rod_len_min	rod_len_max
1	Cast	6	2	86	96
2	Cast	6	2	84	90
3	Cast	6	4	84	90
4	Cast	4	4	86	96
5	Cast	4	4	84	86
6	Cast	4	4	84	88
10	Cast	5	2	84	92

The sidebar on the right shows "Result Grid" is selected. At the bottom of the result grid are buttons for "Apply" and "Revert".

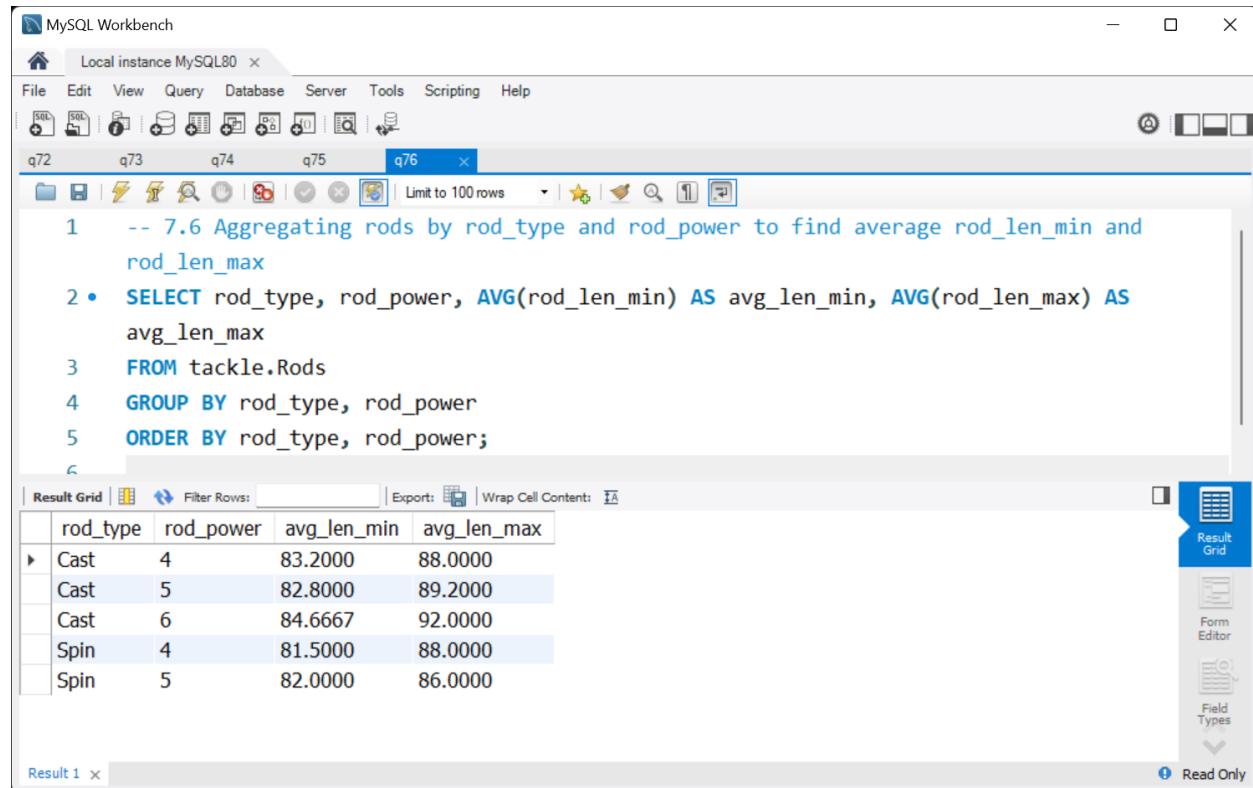
*Note.* Code at <https://github.com/ciiprof0/tackle/blob/main/db/schema/queries/q75.sql>.  
Data at [https://github.com/ciiprof0/tackle/blob/main/data/processed/7\\_5.csv](https://github.com/ciiprof0/tackle/blob/main/data/processed/7_5.csv)

**6. Write an SQL to aggregate the results by using multiple columns in the SELECT clause. Interpret the output.**

The output shows how rods' average minimum and maximum lengths vary based on their type (rod\_type) and power (rod\_power). This information can help identify trends, such as whether certain rod types or power levels tend to have longer or shorter rods on average, offering insights into the design patterns or usage of different rod categories.

**Figure 24**

*Aggregation Using Multiple Columns*



The screenshot shows the MySQL Workbench interface. The query editor window contains the following SQL code:

```

1 -- 7.6 Aggregating rods by rod_type and rod_power to find average rod_len_min and
2 • SELECT rod_type, rod_power, AVG(rod_len_min) AS avg_len_min, AVG(rod_len_max) AS
   avg_len_max
3 FROM tackle.Rods
4 GROUP BY rod_type, rod_power
5 ORDER BY rod_type, rod_power;
6

```

The results grid displays the following data:

rod_type	rod_power	avg_len_min	avg_len_max
Cast	4	83.2000	88.0000
Cast	5	82.8000	89.2000
Cast	6	84.6667	92.0000
Spin	4	81.5000	88.0000
Spin	5	82.0000	86.0000

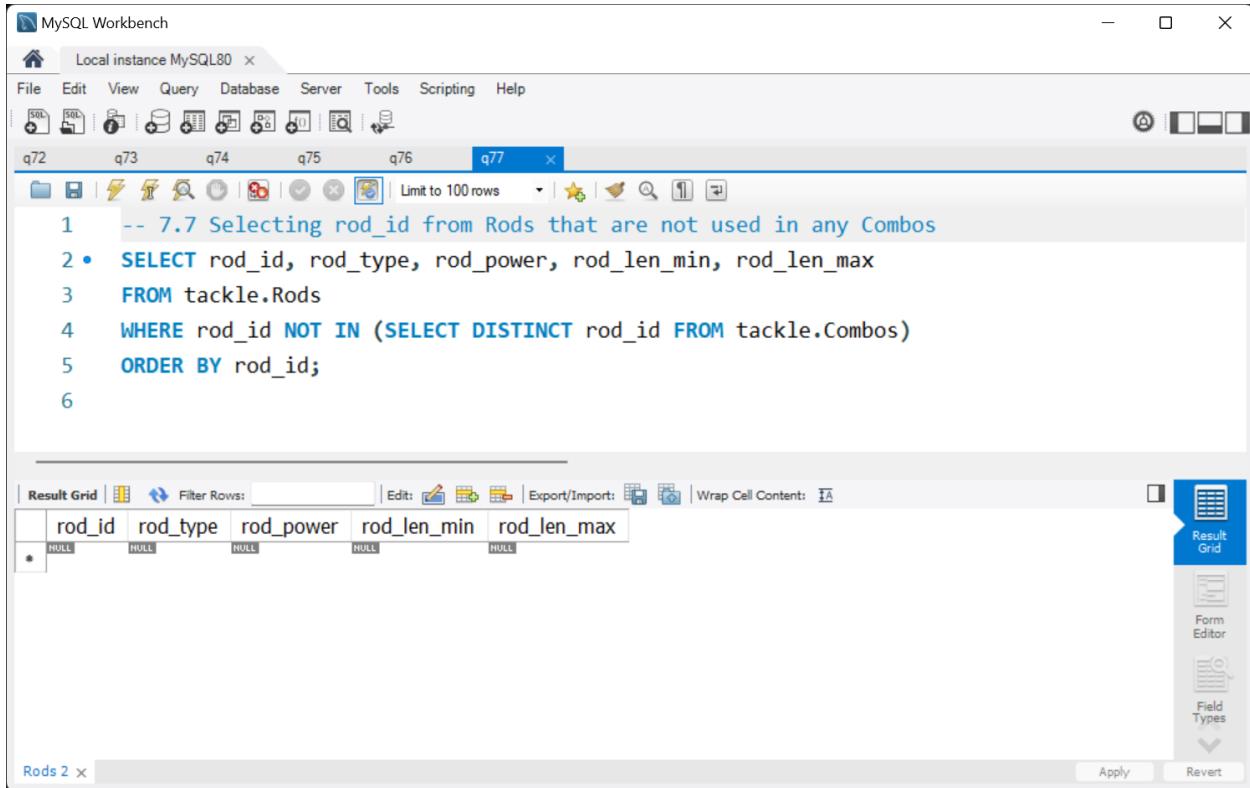
*Note.* Code at <https://github.com/ciiprof0/tackle/blob/main/db/schema/queries/q76.sql>.  
Data at [https://github.com/ciiprof0/tackle/blob/main/data/processed/7\\_6.csv](https://github.com/ciiprof0/tackle/blob/main/data/processed/7_6.csv)

**7. Write a subquery using the NOT IN operator. Show the results and sort the results by key field(s). Interpret the output.**

The query will return a list of rods not associated with combinations in the Combos table. This could indicate available rods that have not been paired with a reel and line in the dataset. Since there are no results as expected, all rods are currently paired in combinations, indicating a complete dataset with no unused rods.

**Figure 25**

*Subquery Using NOT IN*



The screenshot shows the MySQL Workbench interface. The top window is the SQL editor with the title "MySQL Workbench". It contains a query labeled "q77" with the following code:

```

1 -- 7.7 Selecting rod_id from Rods that are not used in any Combos
2 • SELECT rod_id, rod_type, rod_power, rod_len_min, rod_len_max
3   FROM tackle.Rods
4 WHERE rod_id NOT IN (SELECT DISTINCT rod_id FROM tackle.Combos)
5 ORDER BY rod_id;
6

```

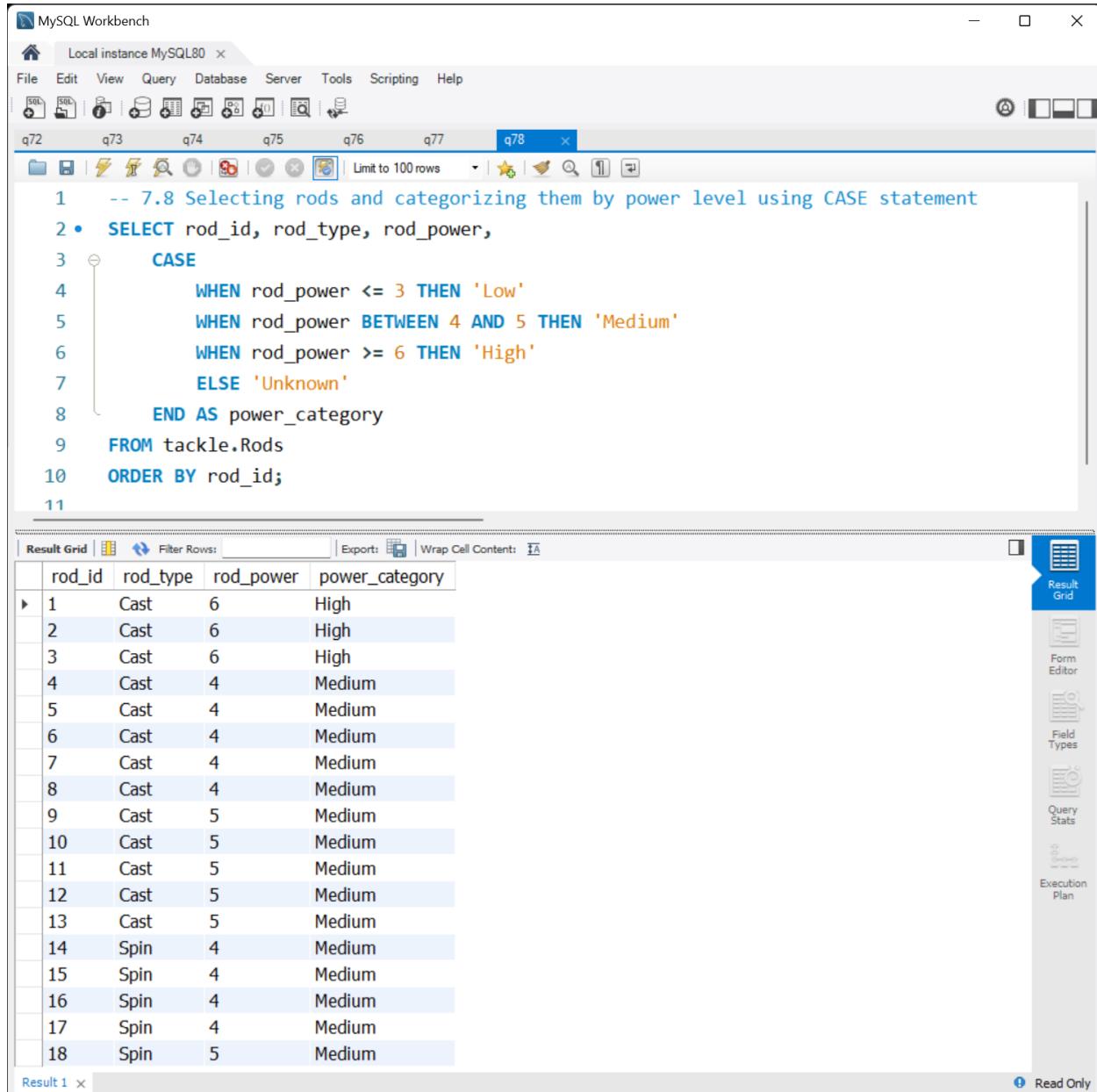
The bottom window is the "Result Grid" showing the results of the query. The table has columns: rod\_id, rod\_type, rod\_power, rod\_len\_min, and rod\_len\_max. There is one row with all values set to NULL.

rod_id	rod_type	rod_power	rod_len_min	rod_len_max
NULL	NULL	NULL	NULL	NULL

*Note.* Code at <https://github.com/ciiprof0/tackle/blob/main/db/schema/queries/q77.sql>.

**8. Write a query using a CASE statement. Show the results and sort the results by key field(s). Interpret the output.**

The results show that most rods fall into the "Medium" power category, with only a few rods classified as "High" power. All "High" power rods are of the "Cast" type with a rod\_power of 6, while the "Medium" power rods include both "Cast" and "Spin" types, with rod\_power values ranging from 4 to 5. This distribution indicates that most of the rods in the dataset are designed with medium power, providing a balance of strength suitable for a wide range of fishing conditions. In contrast, fewer rods are made for higher-power applications.

**Figure 26***Query Using CASE Statement*


The screenshot shows the MySQL Workbench interface with a query editor and a result grid.

**Query Editor:**

```

1 -- 7.8 Selecting rods and categorizing them by power level using CASE statement
2 • SELECT rod_id, rod_type, rod_power,
3     CASE
4         WHEN rod_power <= 3 THEN 'Low'
5         WHEN rod_power BETWEEN 4 AND 5 THEN 'Medium'
6         WHEN rod_power >= 6 THEN 'High'
7         ELSE 'Unknown'
8     END AS power_category
9     FROM tackle.Rods
10    ORDER BY rod_id;
11

```

**Result Grid:**

rod_id	rod_type	rod_power	power_category
1	Cast	6	High
2	Cast	6	High
3	Cast	6	High
4	Cast	4	Medium
5	Cast	4	Medium
6	Cast	4	Medium
7	Cast	4	Medium
8	Cast	4	Medium
9	Cast	5	Medium
10	Cast	5	Medium
11	Cast	5	Medium
12	Cast	5	Medium
13	Cast	5	Medium
14	Spin	4	Medium
15	Spin	4	Medium
16	Spin	4	Medium
17	Spin	4	Medium
18	Spin	5	Medium

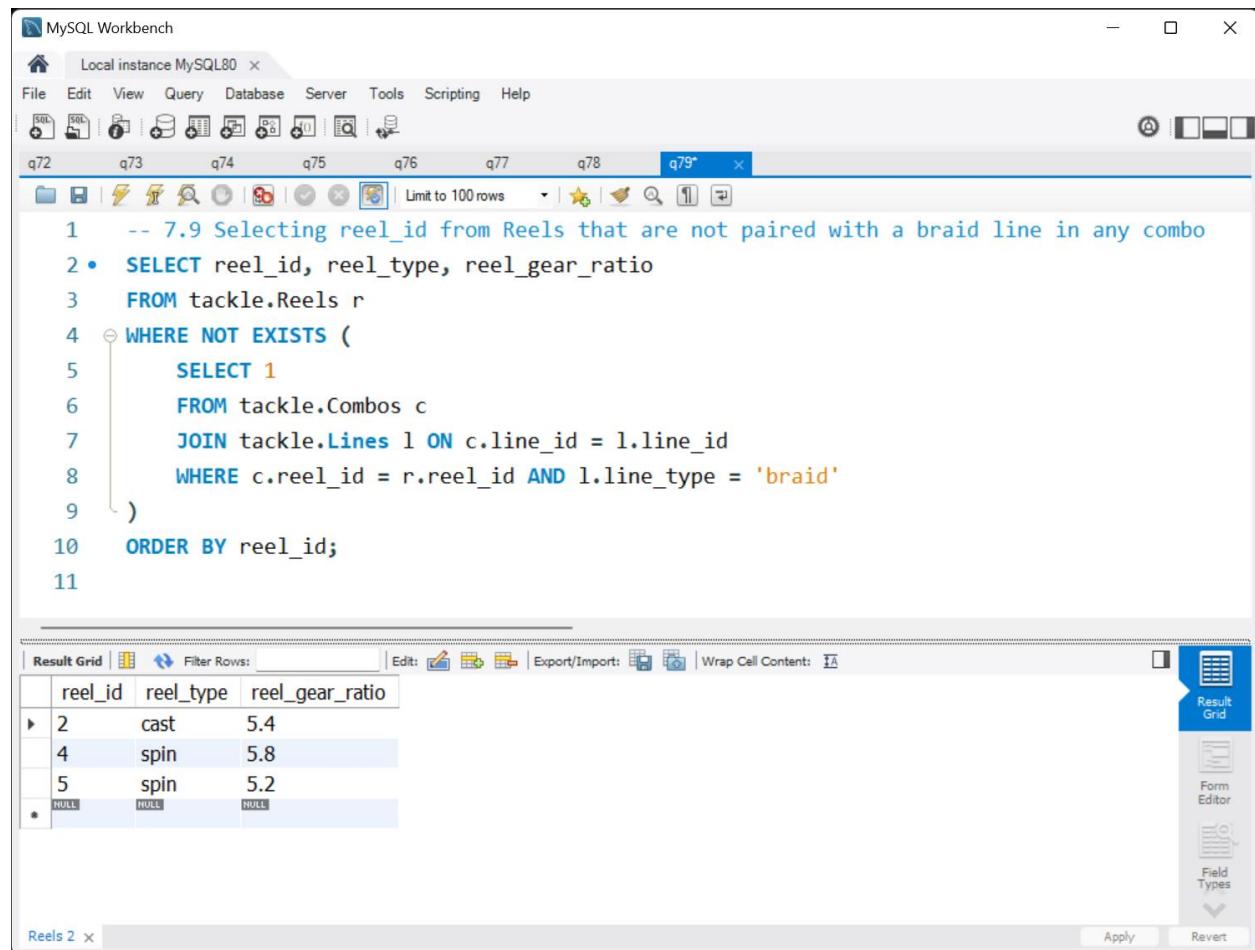
*Note.* Code at <https://github.com/ciiprof0/tackle/blob/main/db/schema/queries/q78.sql>.  
 Data at [https://github.com/ciiprof0/tackle/blob/main/data/processed/7\\_8.csv](https://github.com/ciiprof0/tackle/blob/main/data/processed/7_8.csv)

**9. Write a query using the NOT EXISTS operator. Show the results and sort the results by key field(s). Interpret the output.**

The results show three reels (`reel\_id`'s 2, 4, and 5) that have not been paired with a "braid" line in any combo. This suggests that despite being available, these reels have only been used with other types of lines or not at all in the case of "braid." The absence of "braid" line usage with these reels might indicate a preference for other line types with these specific reels or a potential opportunity to explore new combinations with "braid" lines to expand their usage.

**Figure 27**

*Query Using NOT EXISTS Operator*



The screenshot shows the MySQL Workbench interface. The top window displays a SQL query titled 'q79'. The code is as follows:

```

1 -- 7.9 Selecting reel_id from Reels that are not paired with a braid line in any combo
2 • SELECT reel_id, reel_type, reel_gear_ratio
3   FROM tackle.Reels r
4 WHERE NOT EXISTS (
5     SELECT 1
6       FROM tackle.Combos c
7         JOIN tackle.Lines l ON c.line_id = l.line_id
8       WHERE c.reel_id = r.reel_id AND l.line_type = 'braid'
9   )
10 ORDER BY reel_id;
11

```

The bottom window shows the 'Result Grid' tab with the following data:

reel_id	reel_type	reel_gear_ratio
2	cast	5.4
4	spin	5.8
5	spin	5.2
*	NULL	NULL

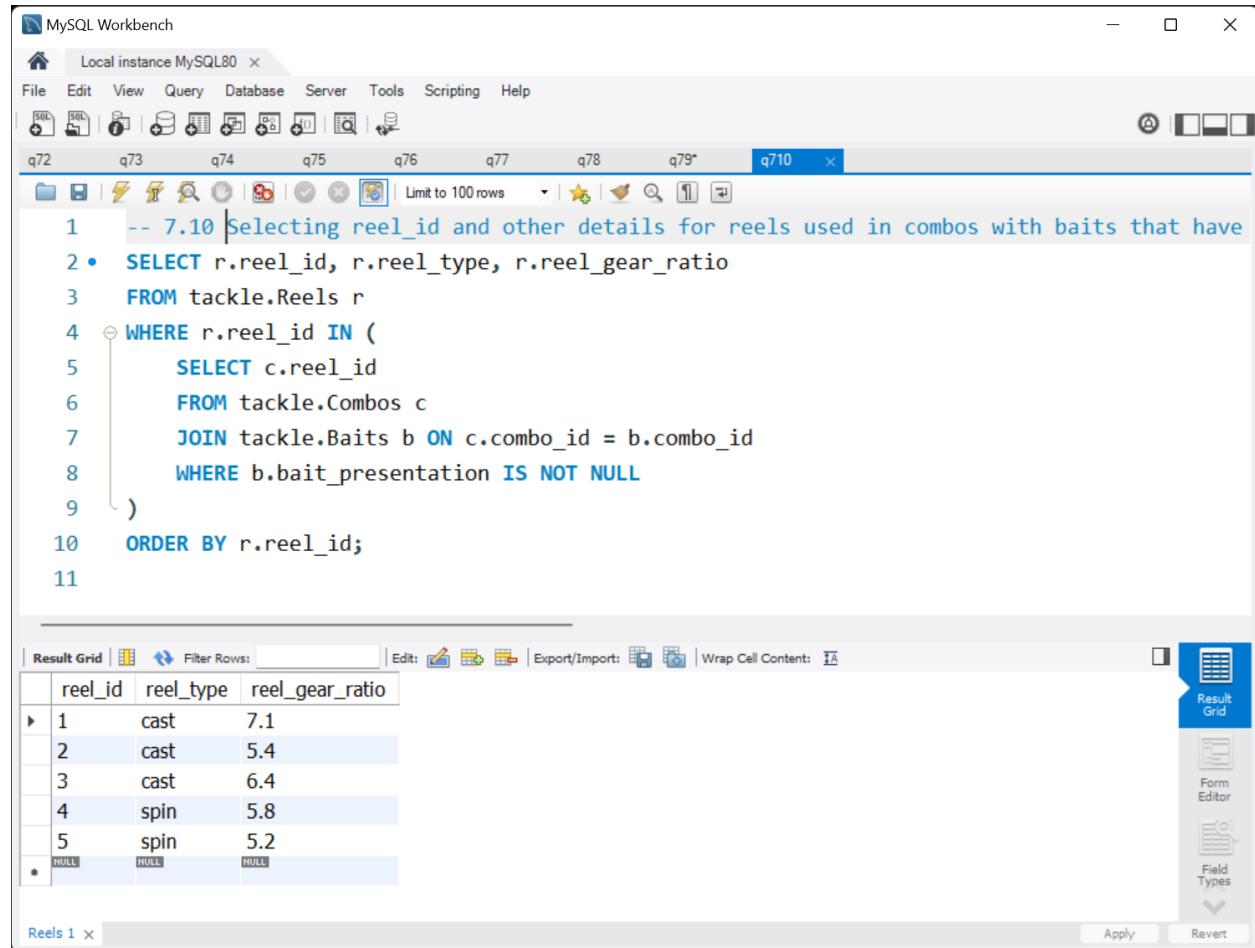
*Note.* Code at <https://github.com/ciiprof0/tackle/blob/main/db/schema/queries/q79.sql>.  
Data at [https://github.com/ciiprof0/tackle/blob/main/data/processed/7\\_9.csv](https://github.com/ciiprof0/tackle/blob/main/data/processed/7_9.csv)

**10. Write a subquery using the NOT NULL operator in the inner query. Show the results and sort the results by key field(s). Interpret the output.**

The results show that all reels (reel\_ids 1, 2, 3, 4, and 5) in the dataset are paired with baits that have a specified bait\_presentation, indicating that these reels are used in combinations where the bait presentation is clearly defined and not missing. This suggests that the dataset regarding bait presentation information for these reels is complete, and there are no instances where a reel is used without a defined bait presentation. This consistency helps ensure that all reels are used effectively with well-documented bait types.

**Figure 28**

### Subquery Using NOT NULL



The screenshot shows the MySQL Workbench interface with a query editor and a result grid.

**Query Editor:**

```

1 -- 7.10 | Selecting reel_id and other details for reels used in combos with baits that have
2 • SELECT r.reel_id, r.reel_type, r.reel_gear_ratio
3   FROM tackle.Reels r
4 WHERE r.reel_id IN (
5   SELECT c.reel_id
6     FROM tackle.Combos c
7     JOIN tackle.Baits b ON c.combo_id = b.combo_id
8     WHERE b.bait_presentation IS NOT NULL
9   )
10 ORDER BY r.reel_id;
11

```

**Result Grid:**

reel_id	reel_type	reel_gear_ratio
1	cast	7.1
2	cast	5.4
3	cast	6.4
4	spin	5.8
5	spin	5.2
*	NULL	NULL

*Note.* Code at <https://github.com/ciiprof0/tackle/blob/main/db/schema/queries/q710.sql>.  
Data at [https://github.com/ciiprof0/tackle/blob/main/data/processed/7\\_10.csv](https://github.com/ciiprof0/tackle/blob/main/data/processed/7_10.csv)