Bacchus Winery Case Study

CSD310 – Database Development & Use

Professor Susan Sampson

BLUE TEAM:

Robert Breutzmann
Carolina Rodriguez
Sara White

Group Introduction

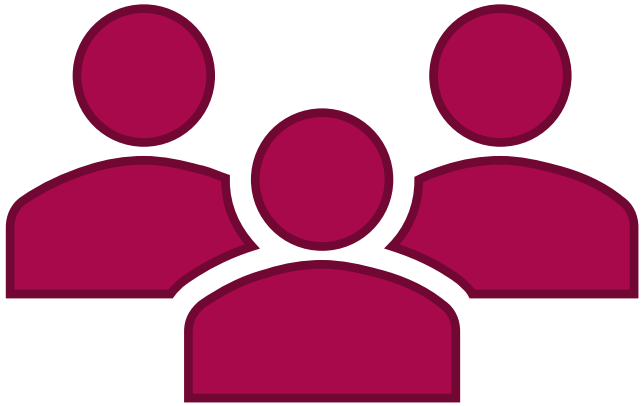
We are part of Team Blue and for our case study we decided to choose the Bacchus Winery Case. Our team members are Robert, Sara and Carolina.

Robert contributed the business rules, the initial ERD, and prepared a report for how many hours each employee worked during the last quarter.

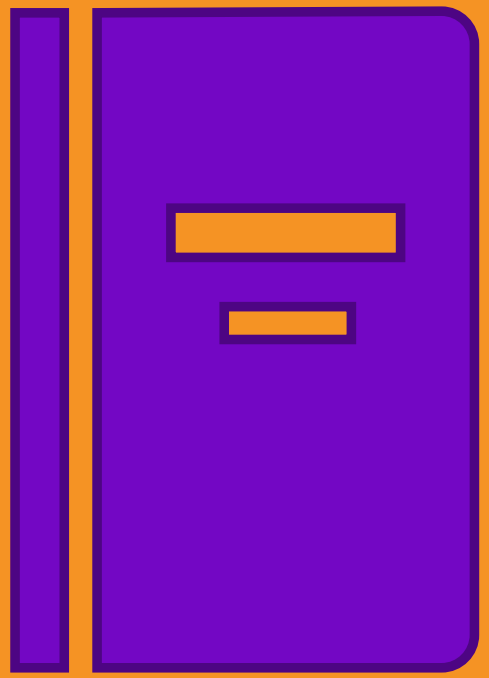
Sara contributed updates to the initial ERD and database, while preparing a report on supplies delivery time and any gap between expected delivery.

Carolina contributed updates to the business rules and initial ERD and prepared a report on which distributor orders which wine, which wine sold, and which didn't.

Together we prepared this presentation and took note on the assumptions we made during this project.

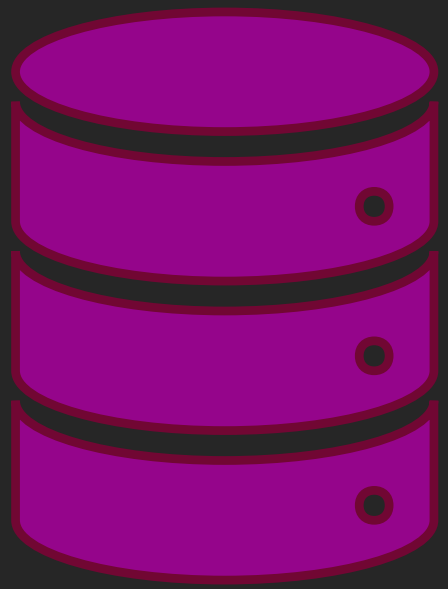


Case Study Overview



- The case study involves helping Bacchus Winery with their operations team by creating a database system to track suppliers, wine production, inventory, distribution and employee hours. The owners need accurate reports on supply deliveries, wine sales and employee work time to make sure their winery is running according to budget.
- Our objective was to design a database structure, create exact reports and ensure data is clear for new decisions.

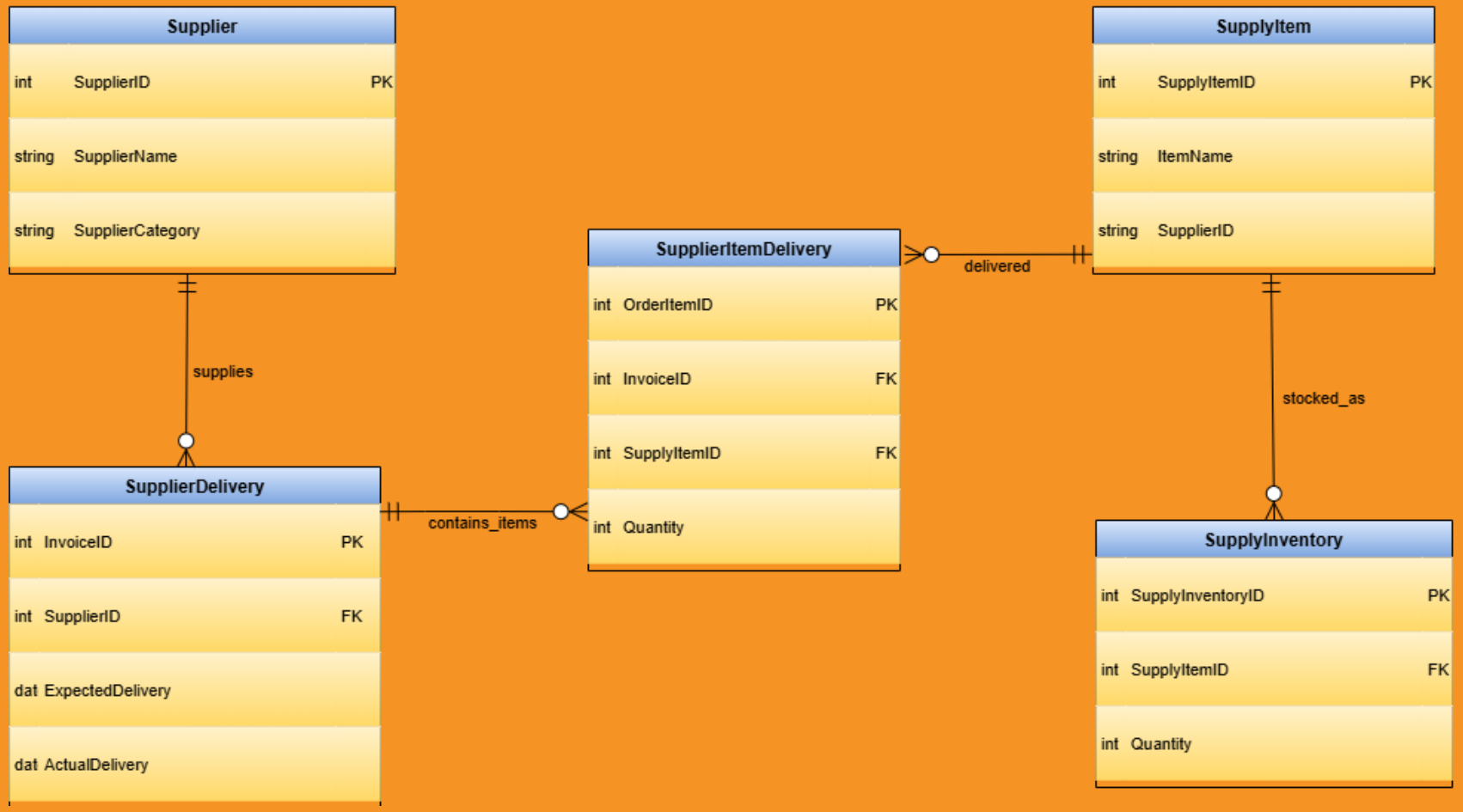
Key Database Requirements



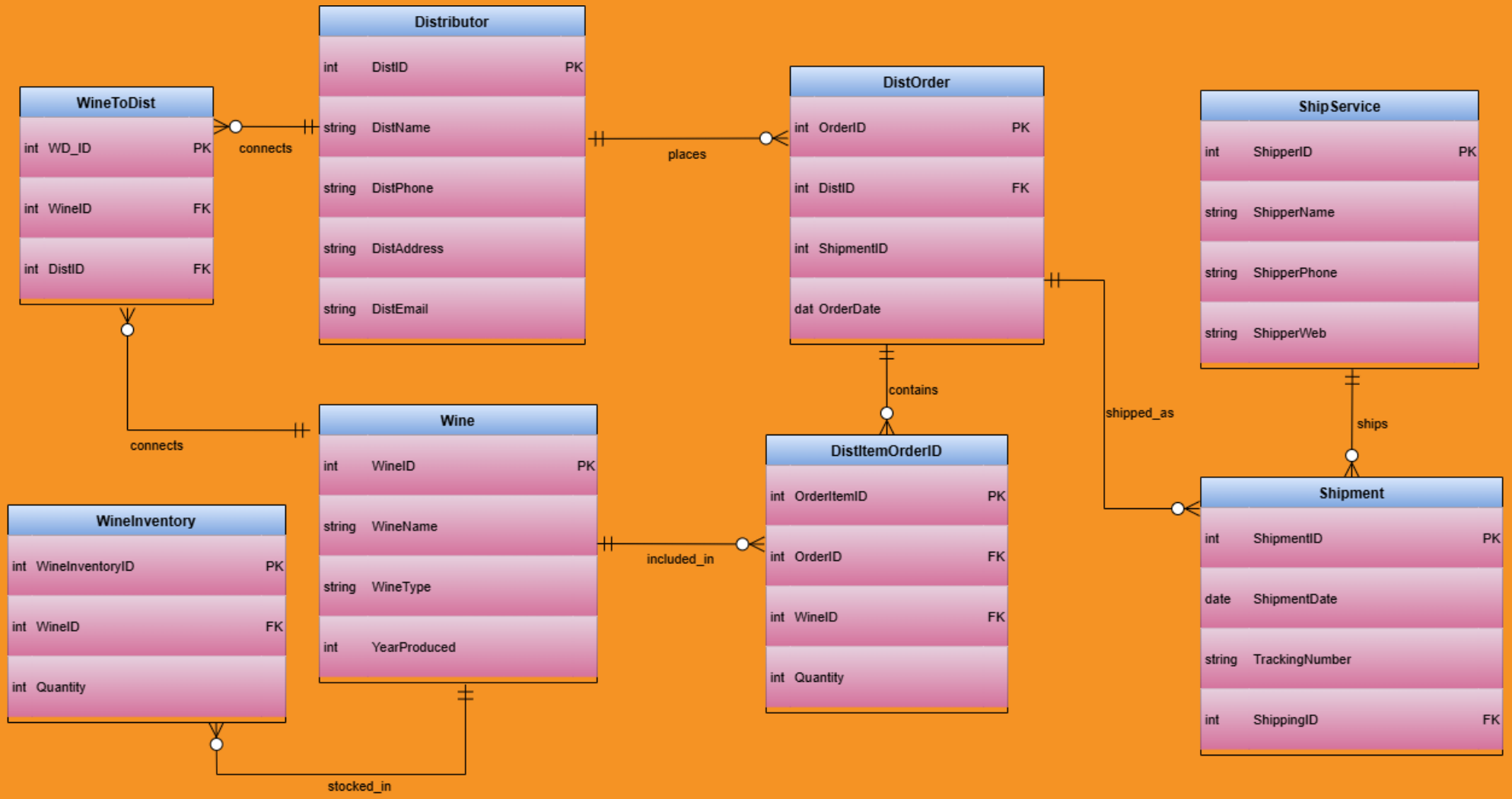
Business Rules:

1. Suppliers provide materials used in wine production, including bottles, corks, labels, boxes, vats, and tubing.
2. Each distributor may carry multiple types of wine produced by Bacchus Winery.
3. Each distributor may carry multiple production years (batches) of each wine type.
4. A distributor may place multiple orders with Bacchus Winery.
5. Each department may employ multiple employees.
6. Each employee may work multiple days and hours, tracked over time.
7. Bacchus Winery produces four types of wine: Merlot, Cabernet, Chablis, and Chardonnay.
8. Each wine type may have multiple batches, distinguished by the year produced.

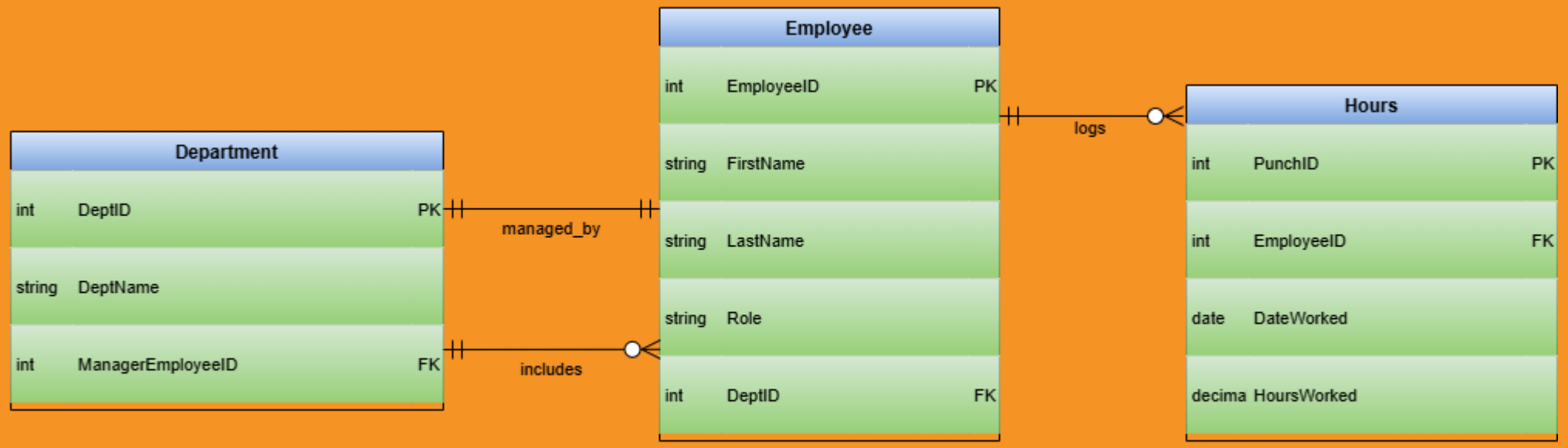
Finalized ERD (1 of 3)



Finalized ERD (2 of 3)



Finalized ERD (3 of 3)



Reports selected for generation (1 of 3)

Wine Distribution (by Distributor)

WineID	WineName	YearProduced	DistID	DistName
1	Merlot	2023	1	Olympus Spirits
2	Cabernet	2023	1	Olympus Spirits
3	Chablis	2023	1	Olympus Spirits
4	Chardonnay	2023	1	Olympus Spirits
2	Cabernet	2023	2	Centaur Wines
3	Chablis	2023	2	Centaur Wines
4	Chardonnay	2023	2	Centaur Wines
2	Cabernet	2023	3	Apollo Imports
4	Chardonnay	2023	3	Apollo Imports
6	Cabernet	2024	3	Apollo Imports
1	Merlot	2023	4	Poseidon Beverage Co
2	Cabernet	2023	4	Poseidon Beverage Co
5	Merlot	2024	4	Poseidon Beverage Co
1	Merlot	2023	5	Minerva Distributors
2	Cabernet	2023	5	Minerva Distributors
3	Chablis	2023	5	Minerva Distributors
4	Chardonnay	2023	5	Minerva Distributors
5	Merlot	2024	6	Vulcan Trade Ltd
6	Cabernet	2024	6	Vulcan Trade Ltd
9	Merlot	2025	6	Vulcan Trade Ltd
10	Cabernet	2025	6	Vulcan Trade Ltd

Total Sold per Wine

WineID	WineName	YearProduced	TotalSold
1	Merlot	2023	135
2	Cabernet	2023	205
3	Chablis	2023	140
4	Chardonnay	2023	180
6	Cabernet	2024	85
5	Merlot	2024	70
9	Merlot	2025	45
10	Cabernet	2025	45

Wines That Haven't Sold

WineID	WineName	YearProduced	TotalSold
7	Chablis	2024	0
11	Chablis	2025	0
8	Chardonnay	2024	0
12	Chardonnay	2025	0

This report provides data in three tables: the first table shows the wines carried by the distributors, the second shows the total amount sold for each type of wine, and the third table shows which wines haven't sold at all.

Use case: Bacchus Winery can use this report for quick insight into which of their wines are best sellers.

Reports selected for generation (2 of 3)

```
================================================================
Supplier Delivery Report - Expected vs Actual
================================================================
InvoiceID | SupplierName         | ExpectedDelivery | ActualDelivery | DaysDifference | DeliveryStatus
-----------------------------------------------------------------------------------------------------
1         | Dionysus Bottlery    | 2025-10-01       | 2025-10-01     | 0              | On Time
2         | Dionysus Bottlery    | 2025-10-01       | 2025-10-01     | 0              | On Time
4         | Hermes Labels & Co   | 2025-10-03       | 2025-10-02     | -1             | Early
5         | Hermes Labels & Co   | 2025-10-03       | 2025-10-02     | -1             | Early
7         | Vulcan Vats & Tubing | 2025-10-05       | 2025-10-20     | 15             | Late
8         | Vulcan Vats & Tubing | 2025-10-05       | 2025-10-25     | 20             | Late
3         | Dionysus Bottlery    | 2025-11-01       | 2025-11-03     | 2              | Late
6         | Hermes Labels & Co   | 2025-11-05       | 2025-11-03     | -2             | Early
9         | Vulcan Vats & Tubing | 2025-11-08       | 2025-11-15     | 7              | Late
```

This report highlights any differences between expected and actual delivery dates from suppliers. The **DaysDifference** column shows how many days the delivery is off from the anticipated delivery date, and the **DeliveryStatus** column states whether the delivery was early or on time.

Use case: Bacchus Winery can use this report to get a quick snapshot into which suppliers are routinely late with deliveries and which suppliers are early or on-time. This could help plan for expected delays with supplies necessary to package and distribute their wines.

Reports selected for generation (3 of 3)

This report was generated to show the average number of hours worked per week by each individual employee. This report uses a dataset for a time period of one month, but could easily be expanded into quarterly or yearly averages.

Three outputs are produced: the first is a list of each employee's ID, followed by their name and average hours worked; the next table shows hours worked by each employee in descending order, and the last organizes this information by the employees last name (alphabetical).

```
[(7, 'Bob', 'Ulrich', Decimal('12.461538')), (17, 'Romulus', 'Rome', Decimal('12.461538')),
 (10, 'Odysseus', 'Ithaca', Decimal('12.442308')), (15, 'Orpheus', 'Thrace', Decimal('12
 .365385')), (8, 'Achilles', 'Myrmidon', Decimal('12.269231')), (22, 'Castor', 'Sparta',
 Decimal('12.230769')), (11, 'Aeneas', 'Trojan', Decimal('12.211538')), (14, 'Jason',
 'Argonaut', Decimal('12.153846')), (19, 'Diomedes', 'Argos', Decimal('12.153846')), (12,
 'Perseus', 'Danae', Decimal('12.038462')), (18, 'Remus', 'Rome', Decimal('12.000000')),
 (21, 'Ajax', 'Salamis', Decimal('11.865385')), (25, 'Meleager', 'Calydon', Decimal('11
 .769231')), (9, 'Hercules', 'Alcides', Decimal('11.750000')), (23, 'Pollux', 'Sparta',
 Decimal('11.538462')), (26, 'Atalanta', 'Arcadia', Decimal('11.519231')), (13, 'Theseus',
 'Athens', Decimal('11.423077')), (16, 'Hector', 'Troy', Decimal('11.076923')), (20,
 'Bellerophon', 'Corinth', Decimal('10.942308')), (24, 'Ariadne', 'Crete', Decimal('10
 .461538'))]
[(9, 'Hercules', 'Alcides', Decimal('11.750000')), (26, 'Atalanta', 'Arcadia', Decimal('11
 .519231')), (14, 'Jason', 'Argonaut', Decimal('12.153846')), (19, 'Diomedes', 'Argos',
 Decimal('12.153846')), (13, 'Theseus', 'Athens', Decimal('11.423077')), (25, 'Meleager',
 'Calydon', Decimal('11.769231')), (20, 'Bellerophon', 'Corinth', Decimal('10.942308')),
 (24, 'Ariadne', 'Crete', Decimal('10.461538')), (12, 'Perseus', 'Danae', Decimal('12
 .038462')), (10, 'Odysseus', 'Ithaca', Decimal('12.442308')), (8, 'Achilles', 'Myrmidon',
 Decimal('12.269231')), (18, 'Remus', 'Rome', Decimal('12.000000')), (17, 'Romulus', 'Rome',
  Decimal('12.461538')), (21, 'Ajax', 'Salamis', Decimal('11.865385')), (22, 'Castor',
 'Sparta', Decimal('12.230769')), (23, 'Pollux', 'Sparta', Decimal('11.538462')), (15,
 'Orpheus', 'Thrace', Decimal('12.365385')), (11, 'Aeneas', 'Trojan', Decimal('12.211538')),
  (16, 'Hector', 'Troy', Decimal('11.076923')), (7, 'Bob', 'Ulrich', Decimal('12.461538'))]
```

Employee's organized by Hours Worked

Last Name	First Name	Hours/Week
Ulrich	Bob	12.46
Rome	Romulus	12.46
Ithaca	Odysseus	12.44
Thrace	Orpheus	12.37
Myrmidon	Achilles	12.27
Sparta	Castor	12.23
Trojan	Aeneas	12.21
Argonaut	Jason	12.15
Argos	Diomedes	12.15
Danae	Perseus	12.04
Rome	Remus	12.0
Salamis	Ajax	11.87
Calydon	Meleager	11.77
Alcides	Hercules	11.75
Sparta	Pollux	11.54
Arcadia	Atalanta	11.52
Athens	Theseus	11.42
Troy	Hector	11.08
Corinth	Bellerophon	10.94
Crete	Ariadne	10.46

Employee's organized by Last Name

Last Name	First Name	Hours/Week
Alcides	Hercules	11.75
Arcadia	Atalanta	11.52
Argonaut	Jason	12.15
Argos	Diomedes	12.15
Athens	Theseus	11.42
Calydon	Meleager	11.77
Corinth	Bellerophon	10.94
Crete	Ariadne	10.46
Danae	Perseus	12.04
Ithaca	Odysseus	12.44
Myrmidon	Achilles	12.27
Rome	Remus	12.0
Rome	Romulus	12.46
Salamis	Ajax	11.87
Sparta	Castor	12.23
Sparta	Pollux	11.54
Thrace	Orpheus	12.37
Trojan	Aeneas	12.21
Troy	Hector	11.08
Ulrich	Bob	12.46

Use case: Bacchus Winery can use this report to budget for payroll, monitor department productivity, and gain business insight based on the hours expended for each department.

Report Generation: Python Code

```
import mysql.connector
from mysql.connector import Error
import dotenv
from dotenv import dotenv_values

import mysql.connector
from mysql.connector import Error
from dotenv import dotenv_values

secrets = dotenv_values("setup.env")

config = {
    "user": secrets["DB_USER"],
    "password": secrets["DB_PASSWORD"],
    "host": secrets["DB_HOST"],
    "database": secrets["DB_NAME"],
    "raise_on_warnings": True
}

def printAlignedData(cursor, data):
    """Print rows in aligned columns, using cursor.description for headers."""

    headers = [desc[0] for desc in cursor.description]

    strData = [[("" if col is None else str(col)) for col in row] for row in data]

    colWidth = []
    for colIDX in range(len(headers)):
        longest = len(headers[colIDX])
        for row in strData:
            longest = max(longest, len(row[colIDX]))
        colWidth.append(longest)

    formatString = " | ".join("{:<" + str(width) + "}" for width in colWidth)

    print(formatString.format(*headers))


    separatorLength = sum(colWidth) + (3 * (len(headers) - 1))
    print("-" * separatorLength)

    if not data:
        print("(No Data to Display)")
        return

    for row in strData:
        print(formatString.format(*row))
```

Our reports were generated and formatted with this initial block of code which securely loads the database credentials utilizing the `dotenv` Python library. This reads sensitive information, like usernames and passwords, from the locally stored `setup.env`.

The config dictionary stores the settings necessary to connect to MySQL. The `printAlignedData()` function extracts the column headers from `cursor.description`, converts the data to strings and then formats the data for readability.

The following slides will contain screenshots of the code unique to each report.

Wines Carried by Distributors and Bestsellers: Python Code

```
def run_query(cursor, query, description):
    print("\n" + "="*70)
    print(description)
    print("-"*70)
    cursor.execute(query)
    rows = cursor.fetchall()
    printAlignedData(cursor, rows)

def main():
    try:
        conn = mysql.connector.connect(**config)
        if not conn.is_connected():
            print("Could not connect to database.")
            return
        cursor = conn.cursor()

        # SQL Wine distribution, ordered by distributor
        wineByDist = """
        SELECT
            w.WineID,
            w.WineName,
            w.YearProduced,
            d.DistID,
            d.DistName
        FROM winetodist wd
        JOIN wine w
            ON wd.WineID = w.WineID
        JOIN distributor d
            ON wd.DistID = d.DistID
        ORDER BY d.DistID, d.DistName
        LIMIT 0, 1000;
        """
        #
        run_query(cursor, wineByDist, "Wine Distribution (by Distributor)")
```

```
        # Total sold per wine
        wineSold = """
        SELECT
            w.WineID,
            w.WineName,
            w.YearProduced,
            SUM(dio.Quantity) AS TotalSold
        FROM Wine w
        JOIN DistItemOrderID dio
            ON w.WineID = dio.WineID
        GROUP BY w.WineID, w.WineName;
        """
        run_query(cursor, wineSold, "Total Sold per Wine")

        # Wines that haven't sold
        wineNOTsold = """
        SELECT
            w.WineID,
            w.WineName,
            w.YearProduced,
            COALESCE(SUM(dio.Quantity), 0) AS TotalSold
        FROM wine w
        LEFT JOIN distitemorderId dio
            ON w.WineID = dio.WineID
        GROUP BY w.WineID, w.WineName, w.YearProduced
        HAVING TotalSold = 0
        ORDER BY w.WineName;
        """
        run_query(cursor, wineNOTsold, "Wines That Haven't Sold")

        cursor.close()
        conn.close()

    except Error as e:
        print("Error: ", e)

if __name__ == "__main__":
    main()
```

Expected vs Actual Supply Delivery: Python Code

```
def main():
    try:
        conn = mysql.connector.connect(**config)
        if not conn.is_connected():
            print("Could not connect to database.")
            return

        cursor = conn.cursor()

        # Report: expected vs actual delivery
        query = """
            SELECT
                sd.InvoiceID,
                s.SupplierName,
                sd.ExpectedDelivery,
                sd.ActualDelivery,
                DATEDIFF(sd.ActualDelivery, sd.ExpectedDelivery) AS DaysDifference,
                CASE
                    WHEN sd.ActualDelivery IS NULL THEN 'Pending'
                    WHEN sd.ActualDelivery < sd.ExpectedDelivery THEN 'Early'
                    WHEN sd.ActualDelivery = sd.ExpectedDelivery THEN 'On Time'
                    WHEN sd.ActualDelivery > sd.ExpectedDelivery THEN 'Late'
                END AS DeliveryStatus
            FROM SupplierDelivery AS sd
            INNER JOIN Supplier AS s
                ON sd.SupplierID = s.SupplierID
            ORDER BY sd.ExpectedDelivery, sd.InvoiceID;
        """
```

```
        cursor.execute(query)
        rows = cursor.fetchall()

        print("=" * 70)
        print("Supplier Delivery Report - Expected vs Actual")
        print("=" * 70)
        printAlignedData(cursor, rows)

        cursor.close()
        conn.close()

    except Error as e:
        print("Error:", e)


if __name__ == "__main__":
    main()
```

Average Weekly Hours for Employees: Python Code

```
def formatData(table):
    reformatedTable = []
    for row in table:
        empID, first, last, hours = row
        roundedHours = float(round(hours, 2))

        reformatedTable.append((last, first, roundedHours))

    return reformatedTable
```

```
def main():
    try:
        conn = mysql.connector.connect(**config)
        if not conn.is_connected():
            print("Could not connect to database.")
            return

        byHour, byName = fetchHoursWorked(conn, "2025-09-01", 13)
        byHour = formatData(byHour)
        byName = formatData(byName)
        print(f"\nEmployee's organized by Hours Worked")
        displayTable(byHour)
        print(f"\nEmployee's organized by Last Name")
        displayTable(byName)

        conn.close()

    except Error as e:
        print("Error: ", e)


if __name__ == "__main__":
    main()
```

Assumptions:

We considered designing a table to keep inventory for the grapes produced by Bacchus Winery, but for the purpose of this case study, we assumed they already have their inventory in place for their grapes that they grow.

We assumed Bacchus Winery would want to keep detailed inventory of wine-making supplies and monitor the timeliness of deliveries from suppliers. Therefore, we created a Supplier table to store supplier information, a **SupplierDelivery** table to track the expected vs. actual delivery dates, and **SupplierItemDelivery** table to track orders with invoices and item information. The **SupplyItem** table contains the specific information on each item that's referenced by the other tables, and the **SupplyInventory** tracks the quantity of those items on hand.

The Wine Distribution portion of the ERD is similarly structured and keeps track of the distributors' information, the wines produced by Bacchus Winery (like the name, year produced, and type of wine) and the current wine inventory of Bacchus Winery. Information, like the year the wine was produced, is useful in determining which specific batches of a type of wine are selling better than others.

The Wine Distribution tables are connected to the distributors' orders and the shipment information for those orders. While our reports don't utilize the **ShipService** table, we felt it was useful to include that table for a fully realized database. For example, we assumed Bacchus Winery might want to monitor which shipping services deliver faster with less issues (like lost or damaged packages).

Challenges and Resolutions

It was challenging to think of a winery's business model from all angles and accurately represent it with our database. We initially didn't account for the need to track the inventory of individual components for a wine business, like bottles, corks, tubing, etc., and had to revise our database design and ERD to include a table to track those items. Other tables, like the ones tracking deliveries from suppliers and the shipments to distributors, were difficult to conceptualize in the beginning because there are so many steps in the supply chain and distribution process.

We thought it would be challenging to find a time that worked for all of us to meet and work on the project together, since we all have busy lives and jobs (and none of us are on the same schedule), however, we were able to meet consistently and communicate with ease.

Thanks for watching our presentation!

References

GeeksforGeeks. (n.d.). SQL CASE statement. https://www.geeksforgeeks.org/sql/sql-case-statement/

Comeau, A. (2017). *MySQL explained*. OSTraining.

W3Resource. (n.d.). *SQL join tables with GROUP BY and ORDER BY*. https://www.w3resource.com/sql/joins/joining-with-group-by-and-order-by.php

W3Schools. (n.d.). MySQL LIMIT clause. https://www.w3schools.com/mysql/mysql_limit.asp

Image credit:

Ivantsov, V. (n.d.). Elegant wine still life with grapes [Photograph]. https://www.pexels.com/photo/elegant-wine-still-life-with-grapes-34815861/