Part 1: Retrieving Data with SELECT

1.1 Retrieving All Expenses:

Write an SQL query to retrieve all data points (columns) from the "Expenses" table.

```
SELECT *
FROM Expenses;
```

1.2 Specific Columns:

Modify your query to select only specific columns relevant to your analysis. For example, you might choose "date," "category," and "amount" to analyze spending patterns by category and date.

```
SELECT date,category,amount
FROM Expenses
ORDER BY date,category;
```

1.3 Filtering by Date Range: Write a query to retrieve expenses charged between a specific date range (e.g., January 1, 2021, to December 15, 2024). Remember to use the appropriate data type for the "date" column when specifying the date range in your query.

```
SELECT date, category, amount
FROM Expenses
WHERE date BETWEEN '2021-01-01' AND '2024-12-15'
ORDER BY date, category;
```

Part 2: Filtering with WHERE Clause

2.1 Filtering by Category: Write a query to find all expenses belonging to a specific category (e.g., "Entertainment").

```
SELECT *
FROM Expenses
WHERE category='Entertainment';
```

2.2 Filtering with Comparison Operators: Find expenses with an amount greater than a certain value (e.g., $50).

```
SELECT *
FROM Expenses
WHERE amount>='50';
```

2.3 Combining Filters (AND):

Refine your query to find expenses that meet multiple criteria. For example, you might search for expenses greater than $75 AND belonging to the "Food" category.

```
SELECT *
FROM Expenses
WHERE category='Food' AND amount=75;
```

2.4 Combining Filters (OR): Modify your query to find expenses belonging to one category or another (e.g., "Transportation" OR "Groceries").

```
SELECT *
FROM Expenses
WHERE category='Transportation' OR category='Groceries';
```

2.5 Filtering with NOT: Write a query to display expenses unrelated to a specific category (e.g., "Rent").

```
SELECT *
FROM Expenses
WHERE category NOT LIKE '%Rent';
```

Part 3: Sorting Retrieved Data

3.1 Sorting by Amount: Write a query to display all expenses sorted by amount in a specific order (e.g., descending order for highest to lowest spending).

```
SELECT *
FROM Expenses
ORDER BY amount DESC;
```

3.2 Sorting by Date and Category:
Modify your query to sort expenses based on multiple columns. For example, you might sort first by date (descending order) and then by category (ascending order) to see recent spending trends by category.

```
SELECT *
FROM Expenses
ORDER BY amount DESC, category ASC;
```

Part 4: Database Upgrade

4.1 Write SQL commands to achieve the following:

- We don't have a table for income yet. Create a table named "Income" with columns for:

  income_id (INT) - Primary Key (auto-increment)

  amount (DECIMAL(10,2)) - NOT NULL

  date (DATE) - NOT NULL

  source (VARCHAR(50)) - NOT NULL

```
CREATE TABLE Income (
income_id INT Primary Key,
amount DECIMAL(10,2) NOT NULL,
date DATE  NOT NULL,
source VARCHAR(50) NOT NULL
);
```

4.2 After creating the "Income" table, you realize you also want to track the income category "source" (e.g., "Salary," "Freelance Work").

- Use ALTER TABLE to add a new column named "category" of type VARCHAR(50).

```
ALTER TABLE income
ADD category VARCHAR(50);
```

4.3 Let's say you decide tracking the income source isn't necessary for now.

- Use ALTER TABLE again to remove the "source" column from the "Income" table.

```
ALTER TABLE Income
DROP COLUMN category;
```

Imagine you no longer need the "Income" table entirely. Experiment how to Use DROP TABLE to permanently remove it from your database.

```
DROP TABLE Income;
```