DATA DRIVEN DECISION USING SQL

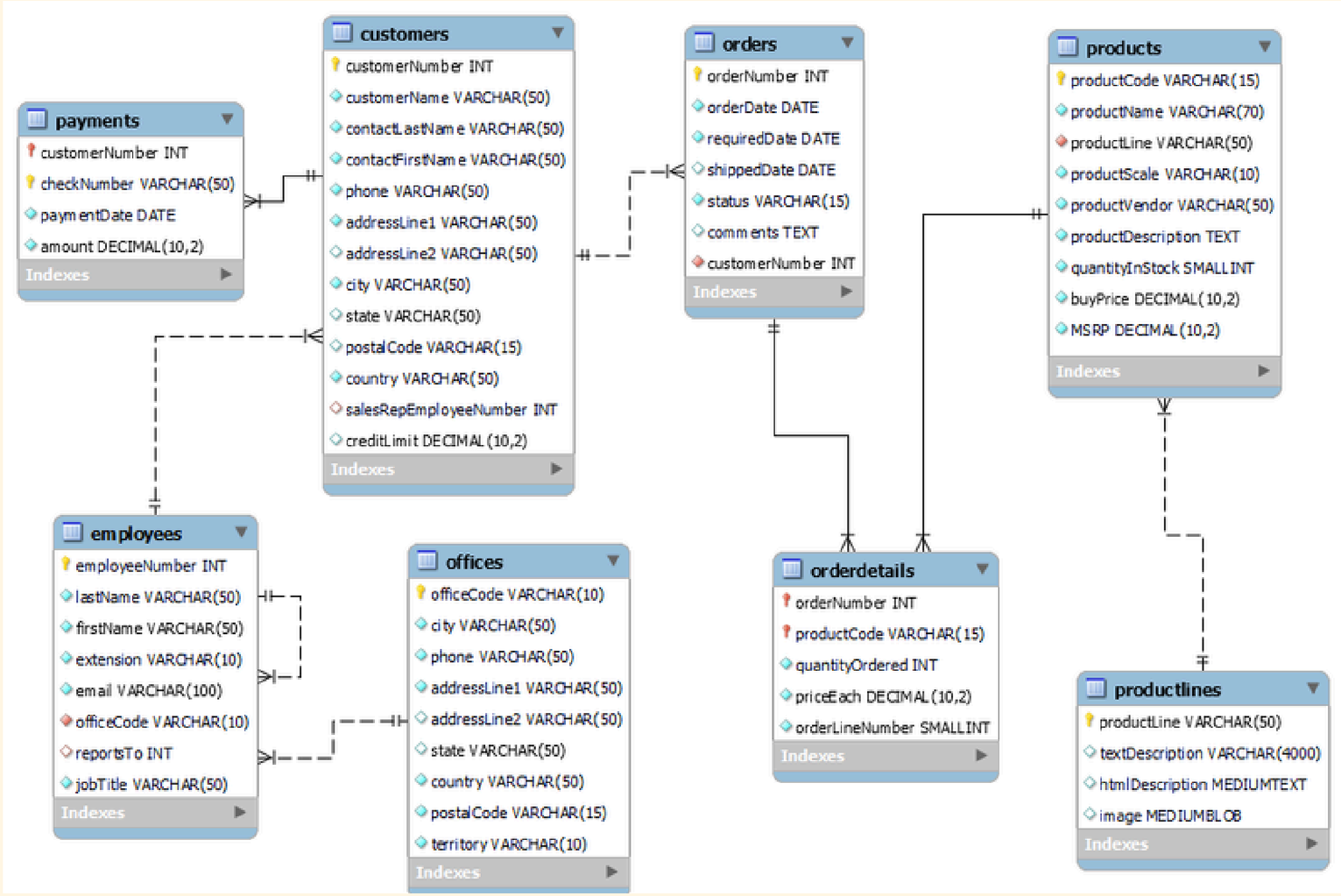
By
Janani

TASK 1.1

- The database captures information related to customers, employees, offices, order details, orders, payments, product lines and products

- The relationships between tables are defined on the basis of foreign key references(E.g. “CustomerNumber in the payments and customers table)

ENTITY-RELATIONSHIP DIAGRAM



TASK 1.1

Query 1: The query aims to retrieve the product codes and their corresponding quantities .

Query 2: This query is used to identify the orders with the longest shipment times among those that have been marked as Shipped in the orders table.

BUSINESS USE

- Improve Customer satisfaction

- Optimize Inventory management

```
select productCode,quantityInStock
from products
order by quantityinstock ASC
limit 5;
```

QUERY 1

	productCode	quantityInStock
▶	S24_2000	15
	S12_1099	68
	S32_4289	136
	S32_1374	178
	S72_3212	414

OUTPUT 1

```
SELECT orderNumber, orderDate, shippedDate - orderdate as shipmenttin
FROM orders
WHERE status = 'Shipped'
order by shipmenttime DESC
limit 5;
```

QUERY 2

orderNumber	orderDate	shipmenttime
10165	2003-10-22	204
10198	2003-11-27	76
10226	2004-02-26	76
10152	2003-09-25	76
10133	2003-06-27	76

OUTPUT 2

TASK 1.2

QUERY 1:To retrieve payment information from the "payments" table and associate each payment with the respective customer.

Query 2:To retrieve information about products and their associated product lines.

BUSINESS USE

- Tracking Customer Payment

- Payment Monitoring

```
SELECT c.customerName, p.paymentDate, p.amount
FROM customers c
JOIN payments p ON c.customerNumber = p.customerNumber
limit 5;
```

QUERY 1

customerName	paymentDate	amount
Atelier graphique	2004-10-19	6066.78
Atelier graphique	2003-06-05	14571.44
Atelier graphique	2004-12-18	1676.14
Signal Gift Stores	2004-12-17	14191.12
Signal Gift Stores	2003-06-06	32641.98

OUTPUT 1

```
SELECT p.productName, p.productCode, pl.productLine
FROM products p
JOIN productlines pl ON p.productLine = pl.productLine
limit 5;
```

QUERY 2

productName	productCode	productLine
1952 Alpine Renault 1300	S10_1949	Classic Cars
1972 Alfa Romeo GTA	S10_4757	Classic Cars
1962 LanciaA Delta 16V	S10_4962	Classic Cars
1968 Ford Mustang	S12_1099	Classic Cars
2001 Ferrari Enzo	S12_1108	Classic Cars

OUTPUT 2

TASK 1.2

QUERY 3:To identify and list customers who have not made any payments.

Query 4: To retrieve information about products and their associated order counts.

```
SELECT c.customerName
FROM customers c
LEFT JOIN payments p ON c.customerNumber = p.customerNumber
WHERE p.customerNumber IS NULL
limit 5;
```

QUERY 3

customerName
Havel & Zbyszek Co
American Souvenirs Inc
Porto Imports Co.
Asian Shopping Network, Co
Natürlich Autos

OUTPUT 3

```
SELECT p.productCode, p.productName, COUNT(od.orderNumber) AS orderCount
FROM products p
LEFT JOIN orderdetails od ON p.productCode = od.productCode
GROUP BY p.productCode, p.productName
limit 5;
```

QUERY 4

productCode	productName	orderCount
S10_1678	1969 Harley Davidson Ultimate Chopper	28
S10_1949	1952 Alpine Renault 1300	28
S10_2016	1996 Moto Guzzi 1100i	28
S10_4698	2003 Harley-Davidson Eagle Drag Bike	28
S10_4757	1972 Alfa Romeo GTA	28

OUTPUT 4

TASK 1.3

QUERY 1: The subquery calculates the total amount for each product in every order. The outer query then calculates the average amount for each product.

Query 2: The subquery calculates the average payment amount. The main query then aggregates the total payments by each customer and filters out only those customers whose total payments exceed the average payment amount.

BUSINESS USE

- Premeium Customers

- Personalized marketing campaign

```sql
SELECT productCode, AVG(orderAmount) as averageOrderAmount
FROM (
    SELECT od.productCode, (od.quantityOrdered * od.priceEach) as orderAmount
    FROM orderdetails od
) AS subquery
GROUP BY productCode;
```

QUERY 1

productCode	averageOrderAmount
S10_1678	3219.920357
S10_1949	6786.355714
S10_2016	3928.529286
S10_4698	6095.928571
S10_4757	4568.725714

OUTPUT 1

```sql
SELECT customerName, SUM(amount) as totalPayments
FROM customers c
JOIN payments p ON c.customerNumber = p.customerNumber
GROUP BY customerName
HAVING totalPayments > (
    SELECT AVG(amount)
    FROM payments
);
```

QUERY 2

customerName	totalPayments
Signal Gift Stores	80180.98
Australian Collectors, Co.	180585.07
La Rochelle Gifts	116949.68
Baane Mini Imports	104224.79
Mini Gifts Distributors Ltd.	584188.24

OUTPUT 2

TASK 1.4

QUERY 1:It provides insights into product sales and cumulative sales totals, with a focus on product lines.

Query 2: It provides information about customer payments, including the running total of payments made by each customer over time.

BUSINESS USE

- Sales analysis

- Decision Making

```
SELECT p.productName, p.productLine, od.quantityOrdered, od.priceEach,
SUM(od.quantityOrdered * od.priceEach) OVER (PARTITION BY p.productLine ORDER BY p.productCode) AS cumulativeTotal
FROM products p
JOIN orderdetails od ON p.productCode = od.productCode
ORDER BY p.productLine, p.productCode
limit 5;
```

QUERY 1

productName	productLine	quantityOrdered	priceEach	cumulativeTotal
1952 Alpine Renault 1300	Classic Cars	26	214.30	190017.96
1952 Alpine Renault 1300	Classic Cars	29	197.16	190017.96
1952 Alpine Renault 1300	Classic Cars	38	205.73	190017.96
1952 Alpine Renault 1300	Classic Cars	37	186.44	190017.96
1952 Alpine Renault 1300	Classic Cars	45	182.16	190017.96

OUTPUT 1

```
SELECT c.customerName, p.paymentDate, p.amount,
SUM(p.amount) OVER (PARTITION BY p.customerNumber ORDER BY p.paymentDate) AS runningTotal
FROM customers c
JOIN payments p ON c.customerNumber = p.customerNumber
ORDER BY c.customerName, p.paymentDate
limit 5;
```

QUERY 2

customerName	paymentDate	amount	runningTotal
Alpha Cognac	2003-07-21	14232.70	14232.70
Alpha Cognac	2003-11-22	33818.34	48051.04
Alpha Cognac	2005-06-03	12432.32	60483.36
Amica Models & Co.	2004-09-04	48298.99	48298.99
Amica Models & Co.	2004-09-19	33924.24	82223.23

OUTPUT 2

TASK 1.5

WHAT?
WHO?
WHY?

INSIGHT ABOUT SALES(Q1(1.1))

- **What:** About top-selling products by analyzing order details.
- **Who:** Sales Department
- **Why:** Helps business to focus on best selling product to maximize sales and Revenue.

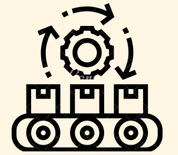
INSIGHT ABOUT CUSTOMER PAYMENT(Q3(1.2))

- **What:** Helps identify customers with overdue payments.
- **Who:** Finance and accounting Department
- **Why:** Helps manage overdue and Credit risks.

TASK 1.5

WHAT?
WHO?
WHY?

INSIGHT ABOUT PRODUCT LINE(Q1(1.4))



- **What:** Helps to analyze performance of product lines and contribution to sales.
- **Who:** Inventory Department
- **Why:** Helps to optimize product line strategies and inventory management.

INSIGHT ABOUT PRODUCT PERFORMANCE(Q2(1.2))

- **What:** Assess the performance of each products within product lines.
- **Who:** Marketing Department
- **Why:** Helps to enhance sales and overall customer satisfaction.

TASK 2.1

Dataset: ENVIRONMENTAL DATA

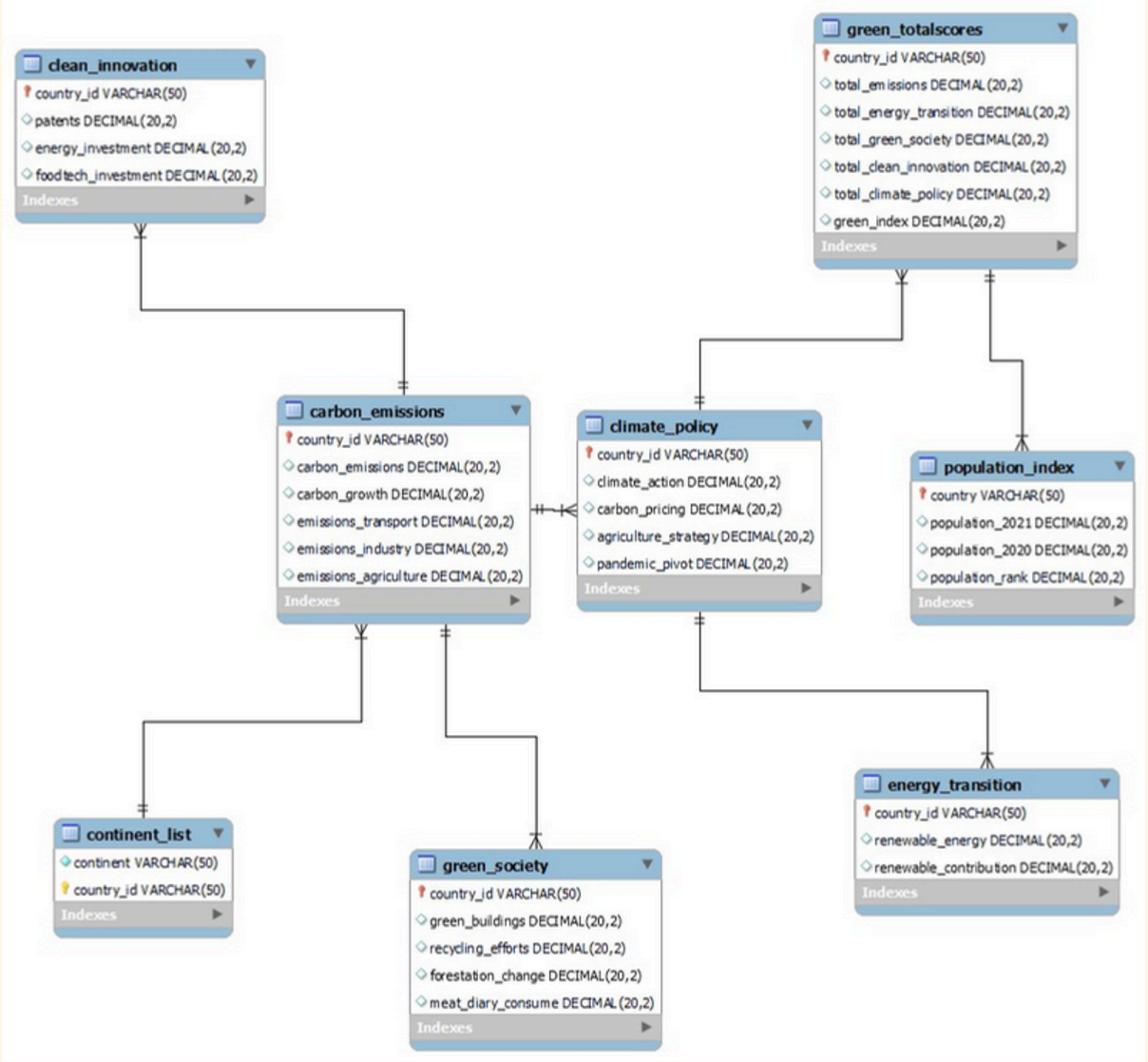
Summary:

Consists of tables such as -

- Clean Innovation
- Carbon emission
- continent list
- green society
- climate policy
- energy transition
- population index
- green score

The relationships between tables are established through foreign keys that reference the “country_id” field

ENTITY-RELATIONSHIP DIAGRAM



TASK 2.2

QUERY 1: Identifying the top 5 countries with the highest green index scores and their respective carbon emissions (db - green scores, carbon emissions)

Query 2: Analysing the impact of renewable energy on the energy transition score across continents (db - continent list, energy transition)

```sql
SELECT
  g.country_id,
  g.green_index,
  c.carbon_emissions
FROM
  green_totalscores AS g
JOIN
  carbon_emissions AS c ON g.country_id = c.country_
ORDER BY
  g.green_index DESC
LIMIT 5;
```

QUERY 1

country_id	green_index	carbon_emissions
Iceland	6.45	10.00
Denmark	6.44	7.12
Norway	6.20	7.00
France	5.98	4.70
Ireland	5.95	7.02

OUTPUT 1

```sql
SELECT
  cl.continent,
  AVG(et.renewable_energy) AS avg_renewable_energy,
  AVG(et.renewable_contribution) AS avg_renewable_contribution
FROM
  energy_transition AS et
JOIN
  continent_list AS cl ON et.country_id = cl.country_id
GROUP BY
  cl.continent;
```

QUERY 2

continent	avg_renewable_energy	avg_renewable_contribution
Africa	4.963333	6.555833
South America	4.488750	4.852500
Oceania	4.055000	3.245000
Europe	3.983750	3.662083
Asia	5.819474	2.553158
North America	4.086000	4.374000

OUTPUT 2

TASK 2.2

QUERY 3: Understanding the relationship between population size and Green Society Measures. (db - population index, green society)

Query 4: Understanding which countries with above average population have a an above average climate action score. (db - population index and climate policy)

BUSINESS USE

- Identifying Evironmental Leaders

- Environmental policy planning

```sql
SELECT
  pi.country,
  pi.population_2021,
  gs.green_buildings + gs.recycling_efforts + gs.forestation_change AS green_society_score
FROM
  population_index AS pi
JOIN
  green_society AS gs ON pi.country = gs.country_id
ORDER BY
  pi.population_2021 DESC;
```

QUERY 3

country	population_2021	green_society_score
China	1445954861.00	9.96
India	1398230990.00	9.16
United States	333612811.00	25.84
Indonesia	277384255.00	7.47
Pakistan	226740747.00	8.88

OUTPUT 3

```sql
SELECT pi.country, cp.climate_action, cp.carbon_pricing, pi.population_2021
FROM climate_policy cp
JOIN population_index pi ON cp.country_id = pi.country
WHERE cp.climate_action > (SELECT AVG(climate_action) FROM climate_policy)
AND pi.population_2021 > (SELECT AVG(population_2021) FROM population_index)
ORDER BY cp.climate_action DESC, pi.population_2021 DESC;
```

QUERY 4

country	climate_action	carbon_pricing	population_2021
United Kingdom	6.00	8.00	68325488.00
France	6.00	8.00	65483524.00
Italy	6.00	7.00	60338805.00
Colombia	6.00	4.00	51411310.00
Canada	6.00	8.00	38185880.00

OUTPUT 4

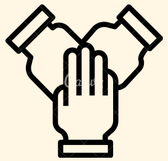
country_id	green_index	carbon_emissions
Iceland	6.45	10.00
Denmark	6.44	7.12
Norway	6.20	7.00
France	5.98	4.70
Ireland	5.95	7.02

TASK 2.3

WHAT?
WHO?
WHY?

country	population_2021	green_society_score
China	1445954861.00	9.96
India	1398230990.00	9.16
United States	333612811.00	25.84
Indonesia	277384255.00	7.47
Pakistan	226740747.00	8.88

COLLABORATION BETWEEN GOVERNMENTS (Q1&3(2.1))



- **What:** Identify countries based on their performance for collaboration
- **Who:** Govts., Policy Makers, Businesses
- **Why:** Impactful climate action with joint efforts

INVESTMENTS IN COUNTRIES WITH HIGH POPULATION (Q3(2.1))

- **What:** Countries with large population and high climate scores
- **Who:** Businesses
- **Why:** Large population countries with high green scores offer eco-friendly investment opportunities and economies of scale

TASK 3.1

Dataset: IBM EMPLOYEES DATASET

Summary:

- Consists of information related to employees working at IBM
- It includes multiple information like their education, employment data, gender, income, satisfaction etc.

ENTITY-RELATIONSHIP DIAGRAM

TASK 3.2

QUERY 1: Provides average monthly income across genders. Useful to analyse income distribution across genders.

Query 2: Rank the employees based on their department and monthly income.

BUSINESS USE

- Pay equity

- Employee Engagement

```
SELECT Gender, AVG(MonthlyIncome) as AverageIncome
FROM employee_dataset_ibm
GROUP BY Gender;
```

QUERY 1

Gender	AverageIncome
Female	6480.3467
Male	6019.1107

OUTPUT 1

```
SELECT Department, EmployeeNumber, MonthlyIncome,
  RANK() OVER (PARTITION BY Department ORDER BY MonthlyIncome DESC) as SalaryRank
FROM employee_dataset_ibm;
```

QUERY 2

Department	EmployeeNumber	MonthlyIncome	SalaryRank
Human Resources	1625	19658	1
Human Resources	1973	19636	2
Human Resources	1550	16437	3
Human Resources	1744	9756	4
Human Resources	2040	8837	5

OUTPUT 2

TASK 3.2

QUERY 3: Creating a view that has important details related to employee. Then, using the view to filter out employees who worked for more than 5 years.

Query 4: Filtering employees based on some filters to see who is most eligible for promotion.

```sql
CREATE VIEW vw_EmployeeWorkDetails AS
SELECT EmployeeNumber, JobRole, YearsAtCompany, TotalWorkingYears
FROM employee_dataset_ibm;

SELECT *
FROM vw_EmployeeWorkDetails
WHERE YearsAtCompany > 5;
```

QUERY 3

EmployeeNumber	JobRole	YearsAtCompany	TotalWorkingYears
1489	Sales Executive	15	16
1497	Sales Executive	10	10
1514	Manufacturing Director	8	8
1520	Manager	14	26
1522	Research Scientist	9	11

OUTPUT 3

```sql
SELECT EmployeeNumber, YearsInCurrentRole, YearsSinceLastPromotion, JobLevel
FROM employee_dataset_ibm
WHERE YearsSinceLastPromotion > 2 AND YearsInCurrentRole > 3 AND JobLevel < 5;
```

QUERY 4

EmployeeNumber	YearsInCurrentRole	YearsSinceLastPromotion	JobLevel
1489	9	10	2
1523	6	14	4
1527	12	5	4
1529	8	3	3
1535	9	8	3

OUTPUT 4

TASK 4

REFLECTION

LEARNING

- Translating raw data into actionable insights

CHALLENGE

- Tailoring recommendations and queries formation

THANK YOU!