Phase 4

Toronto Metropolitan University

Simon Lin (501103322),
Dylan Ha (501056670),
Enes Polat (501061594)

CPS510 - Database Systems

# Point of Sale System for Shopper Drug Marts

4. Simple Queries

## Customer:

Query to list all Customers names and Optimum IDs

```
SELECT name ||  '''s Optimum ID is: ' || optimum_id  AS "All Customer Optimum IDs"
FROM customer;
```

```
All Customer Optimum IDs
-----------------------------------------
Simon Lin's Optimum ID is: 501103322
Dylan Ha's Optimum ID is: 501056670
Enes Polat's Optimum ID is: 501061594
```

## Employee:

Query to list all employees names and their job positions

```
SELECT name, 'Works as a ' || position || ' for Shoppers Drug Mart' AS "Job Description"
FROM employee;
```

```
NAME                   Job Description
---------------------- ------------------------------------------------------------
Ski Betty              Works as a Cashier for Shoppers Drug Mart
Hawk T. Ooah           Works as a Manager for Shoppers Drug Mart
Hugh Mungus            Works as a Owner for Shoppers Drug Mart
```

## Product:

Query to list all product names ascended by their prices with ORDER BY.

```
SELECT SUBSTR(product_name, 1, 30) AS "Products ascending by price", price
FROM product
ORDER BY price;
```

```
Products ascending by price         PRICE
------------------------------ ----------
Apple                                 .99
Chips                                1.99
Milk                                 2.49
Orange Juice                         2.99
```

## Optimum:

Query to list all people who have a total number of points between 10000 and 20000

```
SELECT optimum_id, name || ' has ' || total_points || ' total points'  AS "Points"
FROM optimum
WHERE total_points BETWEEN 10000 AND 20000;

OPTIMUM_ID Points
---------- ----------------------------------------
 501056670 Dylan Ha has 15000 total points
```

## Receipt:

Query to list the sum of all total sales grouped by each payment method.

```
SELECT payment_method AS "Payment_Method", SUM(total_price) AS "Total Sales"
FROM receipt
GROUP BY payment_method;

Payment_Method   Total Sales
---------------- -----------
Debit                   4.98
Cash                   47.18
Credit                 11.94
```

## Transaction:

Query to list the sum of all total sales completed by each employee ID

```
SELECT employee_id AS "Employee ID", COUNT(total_price) AS "Total Sales"
FROM transaction
GROUP BY employee_id;

Employee ID Total Sales
----------- -----------
          1           2
          2           2
          3           3
```

Inventory:
Query to list all products that are out of stock (zero in storage quantity)

```
SELECT SUBSTR(product_name, 1, 30) AS "Out of Stock:"
FROM inventory
WHERE storage_quantity = 0
```

```
Out of Stock:
------------------------------
Potato
```

Source Code:

```
SELECT name ||  "'s Optimum ID is: ' || optimum_id  AS "All Customer Optimum IDs"
FROM customer;

SELECT name, 'Works as a ' || position || ' for Shoppers Drug Mart' AS "Job Description"
FROM employee;

SELECT SUBSTR(product_name, 1, 30) AS "Products ascending by price", price
FROM product
ORDER BY price;

SELECT optimum_id, name || ' has ' || total_points || ' total points'  AS "Points"
FROM optimum
WHERE total_points BETWEEN 10000 AND 20000;

SELECT payment_method AS "Payment_Method", SUM(total_price) AS "Total Sales"
FROM receipt
GROUP BY payment_method;

SELECT employee_id AS "Employee ID", COUNT(total_price) AS "Total Sales"
FROM transaction
GROUP BY employee_id;

SELECT SUBSTR(product_name, 1, 30) AS "Out of Stock:"
FROM inventory
WHERE storage_quantity = 0
```

4. Part 2 - Views and Joins

Receipt:
VIEW to see all recorded **Cashless Payments** (Debit/Credit)

```
CREATE VIEW cashless_payments AS
(SELECT transaction_id, payment_method, product_list, total_price, transaction_date
 FROM receipt
 WHERE payment_method = 'Debit' or payment_method = 'Credit');
```

| | TRANSACTION_ID | PAYMENT_METHOD | PRODUCT_LIST | TOTAL_PRICE | TRANSACTION_DATE |
|---|---|---|---|---|---|
| 1 | 2 | Debit | x1 Milk | 2.49 | 24-SEP-24 |
| 2 | 3 | Credit | x3 Chips | 5.97 | 02-OCT-24 |
| 3 | 5 | Debit | x1 Milk | 2.49 | 24-SEP-24 |
| 4 | 6 | Credit | x3 Chips | 5.97 | 02-OCT-24 |

Product:
VIEW to see all products that are **Produce** (Fruits and Vegetables)

```
CREATE VIEW produce AS
(SELECT PRODUCT_ID, CATEGORY, PRODUCT_NAME, PRICE, SHELF_QUANTITY
 FROM product
 WHERE category = 'Fruit' or category = 'Vegetable');
```

| | PRODUCT_ID | CATEGORY | PRODUCT_NAME | PRICE | SHELF_QUANTITY |
|---|---|---|---|---|---|
| 1 | 5 | Vegetable | Potato | 1 | 20 |
| 2 | 1 | Fruit | Apple | 0.99 | 10 |

Product:
VIEW to see all products behind a **Locked Shelf** (Price is greater than or equal to $100)

```
CREATE VIEW locked_shelf AS
(SELECT PRODUCT_ID, CATEGORY, PRODUCT_NAME, PRICE, SHELF_QUANTITY
 FROM product
 WHERE price >= 100);
```

| | PRODUCT_ID | CATEGORY | PRODUCT_NAME | PRICE | SHELF_QUANTITY |
|---|---|---|---|---|---|
| 1 | 6 | Technology | AirPods Pro | 200 | 10 |

Advanced Queries & Joins:

## Product & Inventory:

Description: List all Dairy products that have a total quantity (on shelves and in inventory) greater than or equal to 50, with a price less than $5.00, ordered by their total quantity.

```
SELECT p.product_id, SUBSTR(p.product_name, 1, 20) AS "Product_Name", p.price, p.shelf_quantity, i.storage_quantity,
      (p.shelf_quantity + i.storage_quantity) AS total_quantity
FROM product p
JOIN inventory i ON p.product_id = i.product_id
WHERE (p.shelf_quantity + i.storage_quantity) >= 50
      AND p.price < 10.00
      AND p.category = 'Dairy'
ORDER BY (p.shelf_quantity + i.storage_quantity) DESC;
```

| PRODUCT_ID | Product_Name | PRICE | SHELF_QUANTITY | STORAGE_QUANTITY | TOTAL_QUANTITY |
|---|---|---|---|---|---|
| 7 | Cheese | 8.99 | 50 | 50 | 100 |
| 4 | Milk | 2.49 | 25 | 50 | 75 |

## Customer & Optimum:

Description: List all Customers with a total number of optimum points between 5000 and 20000, that do not have “Simon” in their name, ordered from greatest points to least points.

```
SELECT c.optimum_id, c.name, o.total_points
FROM customer c
JOIN optimum o ON c.optimum_id = o.optimum_id
WHERE o.total_points > 5000
      AND o.total_points < 20000
      AND c.name NOT LIKE '%Simon%'
ORDER BY o.total_points DESC;
```

| OPTIMUM_ID | NAME | TOTAL_POINTS |
|---|---|---|
| 501056670 | Dylan Ha | 15000 |
| 501061594 | Enes Polat | 9000 |

Source code for A4 Part 2:

**—VIEWS—**

```
CREATE VIEW cashless_payments AS
(SELECT transaction_id, payment_method, product_list, total_price, transaction_date
 FROM receipt
 WHERE payment_method = 'Debit' or payment_method = 'Credit');

CREATE VIEW produce AS
(SELECT PRODUCT_ID, CATEGORY, PRODUCT_NAME, PRICE,
SHELF_QUANTITY
 FROM product
 WHERE category = 'Fruit' or category = 'Vegetable');

CREATE VIEW locked_shelf AS
(SELECT PRODUCT_ID, CATEGORY, PRODUCT_NAME, PRICE,
SHELF_QUANTITY
 FROM product
 WHERE price >= 100);
```

**—JOINS—**

```
SELECT p.product_id, SUBSTR(p.product_name, 1, 20) AS "Product_Name", p.price,
p.shelf_quantity, i.storage_quantity,
    (p.shelf_quantity + i.storage_quantity) AS total_quantity
FROM product p
JOIN inventory i ON p.product_id = i.product_id
WHERE (p.shelf_quantity + i.storage_quantity) >= 50
    AND p.price < 10.00
    AND p.category = 'Dairy'
ORDER BY (p.shelf_quantity + i.storage_quantity) DESC;

SELECT c.optimum_id, c.name, o.total_points
FROM customer c
JOIN optimum o ON c.optimum_id = o.optimum_id
WHERE o.total_points > 5000
    AND o.total_points < 20000
    AND c.name NOT LIKE '%Simon%'
ORDER BY o.total_points DESC;
```

```
#!/bin/sh
#export LD_LIBRARY_PATH=/usr/lib/oracle/12.1/client64/lib
sqlplus64
"s1lin/09183322@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=oracle.scs.r
yerson.ca)(Port=1521))(CONNECT_DATA=(SID=orcl)))" <<EOF

SELECT name || "'s Optimum ID is: ' || optimum_id AS "Total points and Optimum IDs"
FROM customer
UNION
SELECT name || ' has ' || total_points || ' total points' AS "Total points and Optimum IDs"
FROM optimum
WHERE total_points BETWEEN 8000 AND 20000;

exit;
EOF
```