

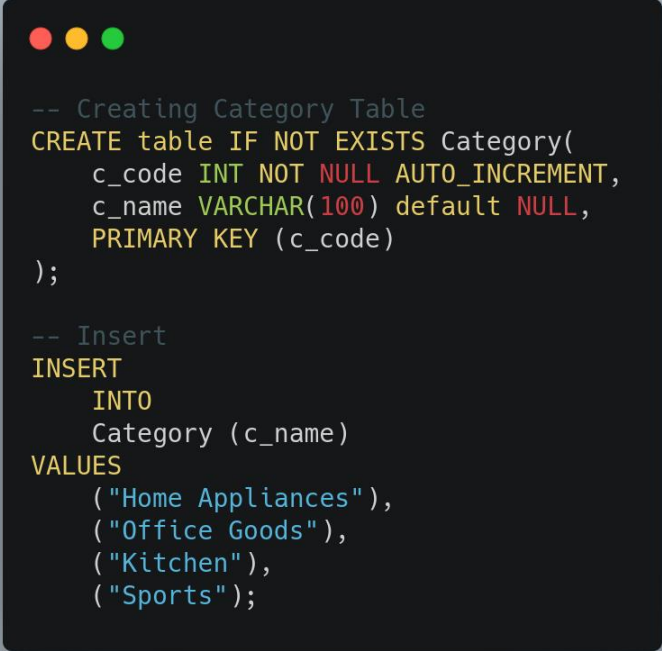
# SQL Concepts | Date – 11/02/2021

Spring AU 2021 – February ( Batch – 2 )

Name – Abhinav Sharma

Email – [abhinav.sharma@accolitedigital.com](mailto:abhinav.sharma@accolitedigital.com)

## CREATION OF TABLES AND INSERTION OF DATA



```
-- Creating Category Table
CREATE table IF NOT EXISTS Category(
  c_code INT NOT NULL AUTO_INCREMENT,
  c_name VARCHAR(100) default NULL,
  PRIMARY KEY (c_code)
);

-- Insert
INSERT
  INTO
    Category (c_name)
VALUES
  ("Home Appliances"),
  ("Office Goods"),
  ("Kitchen"),
  ("Sports");
```

```
-- Creating Product table
CREATE table IF NOT EXISTS Product(
  p_code INT NOT NULL AUTO_INCREMENT,
  p_name VARCHAR(100) default NULL,
  unit_price DOUBLE(12,2) default 0.0,
  category_assigned INT Not Null,
  PRIMARY KEY (p_code),
  FOREIGN KEY (category_assigned) REFERENCES Category(c_code)
);

-- Insert
INSERT
  INTO
    Product (p_name, unit_price, category_assigned)
VALUES
  ("Heater", 900.00, 1),
  ("Table", 20900.45, 1),
  ("Laptop", 300.56, 2),
  ("Mouse", 890, 2),
  ("Utencils", 1200.00, 3),
  ("Gas Stove", 2200.45, 3),
  ("Bat", 3300.56, 4),
  ("Football", 8490, 4);
```

```
-- Creating Location
CREATE table IF NOT EXISTS Location(
    loc_code INT NOT NULL AUTO_INCREMENT,
    loc_name VARCHAR(100) default NULL,
    PRIMARY KEY (loc_code)
);

-- Insert
INSERT
    INTO
    Location (loc_name)
VALUES
    ("India"),
    ("USA"),
    ("Canada"),
    ("Ontario"),
    ("Indiana"),
    ("America"),
    ("Netherland"),
    ("Morocco");
```

```
-- Creating SalesExecutive
CREATE table IF NOT EXISTS SalesExecutive(
  employee_id INT NOT NULL AUTO_INCREMENT,
  name VARCHAR(100) default NULL,
  dob DATE NOT NULL,
  gender VARCHAR(20) NOT NULL,
  CHECK (gender IN('M', 'F', 'OTHERS')),
  mob_no BIGINT(10) NOT NULL,
  location INT NOT NULL,
  PRIMARY KEY (employee_id),
  FOREIGN KEY (location) REFERENCES Location(loc_code)
);

-- Insert
INSERT
  INTO
    SalesExecutive (name, dob, gender, mob_no, location)
VALUES
  ("Abhik", "1998-07-07", "M", 7909948987, 1),
  ("Ana", "1998-07-07", "F", 7909949834, 2),
  ("Ishaan", "1998-07-07", "M", 7909944587, 3),
  ("Lee", "1998-07-07", "F", 7909944387, 8),
  ("Muskan", "1998-07-07", "F", 7909338987, 5),
  ("Kiran", "1998-07-07", "M", 7909933987, 6),
  ("Kishan", "1998-07-07", "M", 7909945687, 7),
  ("Dinesh", "1998-07-07", "M", 7909942487, 2);
```

```
-- Creating Customer
CREATE table IF NOT EXISTS Customer(
  c_id INT NOT NULL AUTO_INCREMENT,
  name VARCHAR(100) default NULL,
  dob DATE NOT NULL,
  gender VARCHAR(20) NOT NULL,
  CHECK (gender IN('M', 'F', 'OTHERS')),
  mob_no BIGINT(10) NOT NULL,
  PRIMARY KEY (c_id)
);

-- Insert
INSERT
  INTO
    Customer (name, dob, gender, mob_no)
VALUES
  ("Abhikansh", "1998-07-07", "M", 9909948987),
  ("Anamika", "1998-08-07", "F", 9909949834),
  ("Haanik", "1998-08-07", "M", 9909944587),
  ("Leela", "1998-01-07", "F", 9909944387),
  ("Nivedita", "1995-07-07", "F", 9909338987),
  ("Anant", "1998-05-07", "M", 9909933987),
  ("Ajitesh", "1928-07-07", "M", 9909945687),
  ("Ritwik", "1993-07-07", "M", 9909942487);
```

```

-- Creating PurchaseSummary
CREATE table IF NOT EXISTS PurchaseSummary(
  purchase_id INT NOT NULL AUTO_INCREMENT,
  sales_exec INT NOT NULL,
  product INT NOT NULL,
  customer INT NOT NULL,
  date_of_purchase DATE NOT NULL,
  no_of_units BIGINT NOT NULL,
  PRIMARY KEY (purchase_id),
  FOREIGN KEY (sales_exec) REFERENCES SalesExecutive(employee_id),
  FOREIGN KEY (product) REFERENCES Product(p_code),
  FOREIGN KEY (customer) REFERENCES Customer(c_id)
);

-- Insert
INSERT INTO
  PurchaseSummary (sales_exec, product, customer, date_of_purchase, no_of_units)
VALUES
  (3, 2, 2, "2021-02-06", 990329),
  (3, 2, 1, "2021-02-06", 992049),
  (3, 2, 2, "2021-02-06", 992309),
  (3, 1, 1, "2021-02-07", 9909),
  (3, 1, 2, "2021-02-07", 9909),
  (3, 2, 1, "2021-02-07", 99209),
  (3, 2, 2, "2021-02-07", 990329),
  (3, 2, 1, "2021-02-07", 992049),
  (3, 2, 2, "2021-02-07", 992309),
  (2, 2, 1, "2021-01-01", 4587),
  (3, 3, 2, "2021-01-01", 4387),
  (4, 4, 3, "2021-02-01", 87),
  (5, 5, 4, "2021-02-02", 7),
  (5, 6, 5, "2021-02-02", 87),
  (6, 4, 6, "2021-02-02", 87),
  (7, 5, 1, "2021-01-01", 687),
  (8, 2, 7, "2021-01-01", 487),
  (7, 7, 8, "2021-02-01", 99),
  (8, 8, 1, "2021-01-01", 990);

```

## QUERY AND OUTPUT

1.

```
-- QUERY

Use 'Au Session';

-- Write a query to retrieve the most sold product per day in a specific location in the last week. You
can pick the location of your choice.

SELECT
    pro.p_code,
    pro.p_name,
    joined.date_of_purchase,
    joined.total_units_sold,
    joined.total_times_sold,
    pro.category_assigned
FROM
    Product as pro
INNER JOIN (
    SELECT
        ps.product,
        ps.date_of_purchase,
        SUM(ps.no_of_units) as total_units_sold,
        Count(*) as total_times_sold,
        DENSE_RANK() OVER(PARTITION BY ps.date_of_purchase
ORDER BY
        SUM(ps.no_of_units) DESC) d_rank
FROM
    PurchaseSummary AS ps
INNER JOIN SalesExecutive AS se ON
    se.employee_id = ps.sales_exec
WHERE
    se.location = 1
AND WEEK(ps.date_of_purchase) = WEEK(NOW()) - 1
GROUP BY
    ps.product ,
    ps.date_of_purchase
ORDER BY
    ps.date_of_purchase,
    SUM(ps.no_of_units) DESC) as joined ON
    joined.product = pro.p_code
where
    joined.d_rank = 1;

-- OUTPUT
```

	p_code	p_name	date_of_purchase	total_units_sold	total_times_sold	category_assigned
2	Table	2021-02-07	3073896	4	1	
1	Heater	2021-02-08	9909	1	1	

## SCREENSHOT

The screenshot shows the DBeaver 7.3.4 interface. The SQL Editor on the right contains the same query as the first image. The Output pane at the bottom displays the query results in a table format, matching the output shown in the first image. The table has columns: p\_code, p\_name, date\_of\_purchase, total\_units\_sold, total\_times\_sold, and category\_assigned. The results are:

	p_code	p_name	date_of_purchase	total_units_sold	total_times_sold	category_assigned
2	Table	2021-02-07	3,073,896	4	1	
1	Heater	2021-02-08	9,909	1	1	

2.

```
-- QUERY

Use 'Au Session';

-- Write a query to list all the salesperson's details along with the count of products sold by them
(if any) till the current date.

SELECT
se2.employee_id,
se2.name,
se2.gender,
se2.dob,
se2.location,
se2.mob_no,
joined.total_products_sold
FROM
SalesExecutive AS se2
INNER JOIN (
SELECT
se.employee_id,
Count(*) AS total_products_sold
FROM
SalesExecutive AS se
LEFT JOIN PurchaseSummary AS ps ON
ps.sales_exec = se.employee_id
GROUP BY
se.employee_id) as joined ON
se2.employee_id = joined.employee_id;

-- OUTPUT
```

employee_id	name	gender	dob	location	mob_no	total_products_sold
1	Abhik	M	1998-07-07	1	7909948987	11
2	Ana	F	1998-07-07	2	7909949834	2
3	Ishaan	M	1998-07-07	3	7909944587	11
4	Lee	F	1998-07-07	8	7909944387	2
5	Muskan	F	1998-07-07	5	7909338987	4
6	Kiran	M	1998-07-07	6	790933987	2
7	Kishan	M	1998-07-07	7	7909945687	4
8	Dinesh	M	1998-07-07	2	7909942487	4

## SCREENSHOT OF OUTPUT

