# Lab - 9 DB_Project_Assignment_ERD Normalization and Schema Refinement, DDL Codes and SQL Queries

**Name:** Ronit Jain (202001081)

Nipun Shah (202001096)

**Group No:** 2

**Section No:** 6

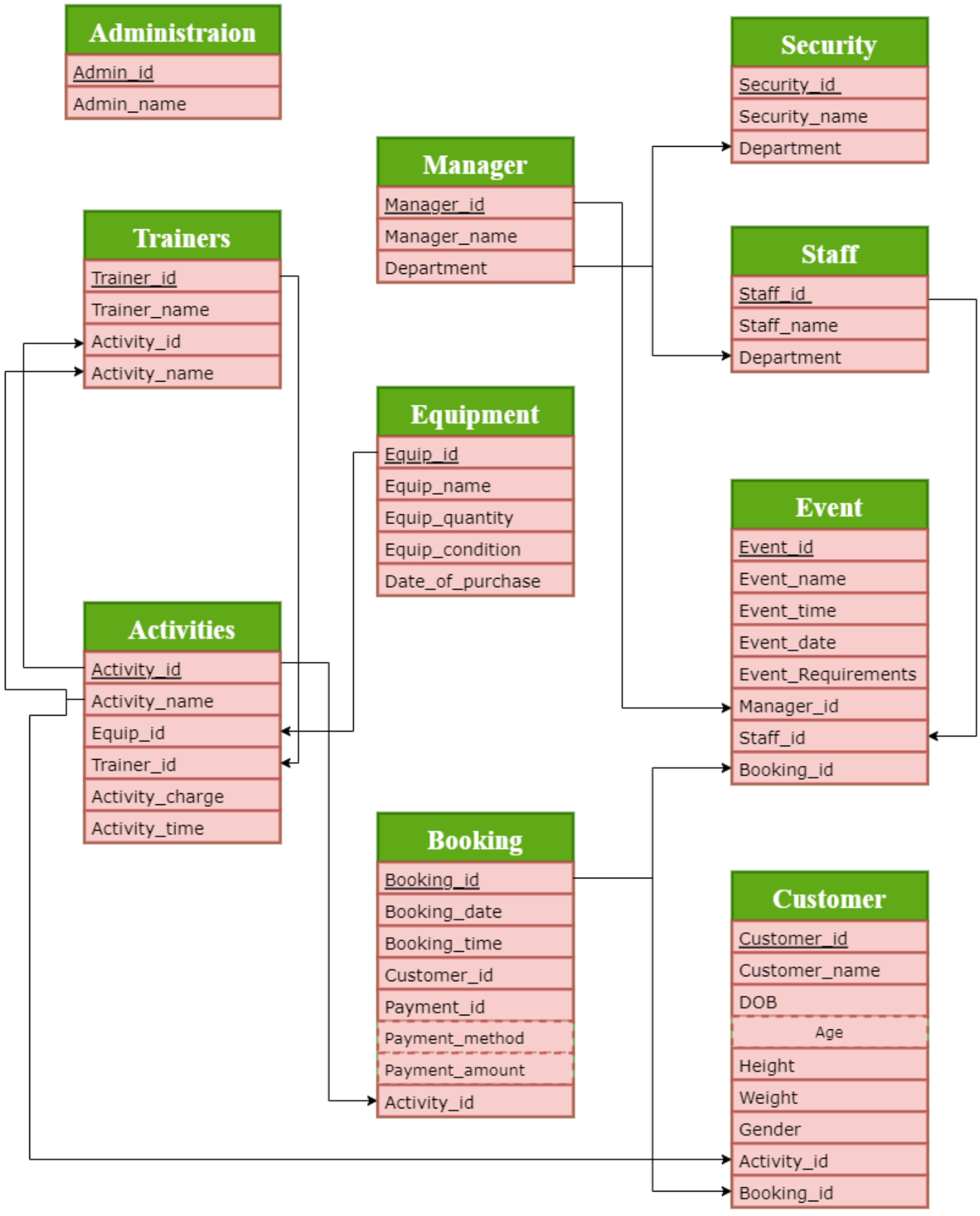
**Team Id:** 6.1

## Case Study

### Beach Activity Management System:

All beach activity-related records should be created in this database. The beach-related activities like scuba diving, snorkeling, paragliding, flyboarding, windsurfing, and so on. Precautionary things like helmets, swimsuits, quality shoes, gloves, etc., for particular activities, must be appropriately managed for all customers. The other things related to taking pictures and videos are also included in this database.

# Normalization & Schema Refinement:



## 1. Administration (<u>Admin_id</u>, Admin_name)

**Primary Key:** Admin_id

**Dependencies:** Admin_id -> Admin_name

**Partial Key Dependency:** None
**Transitive Dependency:** None

**Redundancies:** In Admin_name, it is possible that the same name is associated with a different Admin_id.

**Anomalies:**
Insert – None.
Update – None.
Delete – None.

Every attribute in this schema is single-valued (scalar), already in 1NF. There is no partial dependency here, so it is in 2NF.

**2NF Redundancies:** None
Since there is no transitive dependency, it is also in 3NF. Thus, our final schema remains the same.

For a relation to be in **BCNF**,

a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super key.

Hence, this relation is in BCNF as well.

## 2. Manager (Manager_id, Manager_name, department)

**Primary Key:** Manager_id

**Dependencies:**
Manager_id -> Manager_name,
Manager_id -> Department

**Partial Key Dependency:** None

**Transitive Dependency:** None

**Redundancies:** In Manager_name, it is possible that the same name is associated with a different Manager_id. It might be possible that two manager_id are associated with the same department.

**Anomalies:**
Insert – None.
Update – None.
Delete – None.

In this schema, every attribute is single-valued (scalar), making it already in 1NF. There is no partial dependency here, so it is in 2NF.

**2NF Redundancies:** None
Since there is no transitive dependency, it is also in 3NF. Thus, our final schema remains the same.

For a relation to be in **BCNF**,

a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super key.

Hence, this relation is in BCNF as well.

3. **Customer (Cutomer_id, Customer_name, DOB, age, Height, Weight, Gender, Activtiy_id, Booking_id)**

**Primary Key:** Customer_id

**Foreign Key:** Activity_id, Booking_id

**Dependencies:**
Customer_id -> Customer_name,
Customer_id -> DOB,
Customer_id -> Age,
DOB -> Age,
Customer_id -> Height,
Customer_id -> Weight,
Customer_id -> Gender,

**Partial Key Dependency:** None

**Transitive Dependency:** Age is derived from DOB, and DOB is derived from Customer_id. We will drop the age attribute as it can be derived from the date of birth (DOB).

**Redundancies:** In Customer_name, it is possible that the same name is associated with a different Cuistomer_id.

**Anomalies:**
Insert – None.
Update – A customer's DOB cannot be updated once set.
Delete – Once assigned, one cannot delete the DOB or Activity_id of a particular customer.

In this, we drop booking_id, as in the booking table there is customer_id through which we can get all the booking details.

In this schema, every attribute is single-valued (scalar), making it already in 1NF. There is no partial dependency here, so it is in 2NF.

**2NF Redundancies:** None

Since there is no transitive dependency, it is also in 3NF. Thus, our final schema remains the same.

For a relation to be in **BCNF**,

a. It should be in the third standard form (3NF).
b. For any dependency A $\rightarrow$ B, A must be a super key.

Hence, this relation is in BCNF as well.

4. **Booking (Booking_id, Booking_date, Booking_time, Customer_id, Payment_id, Payment_method, Payment_amount, Activity_id)**

**Primary Key:** Booking_id

**Foreign Key:** Customer_id, Activity_id

**Dependencies:**
Booking_id -> Booking_date,
Booking_id -> Booking_time,
Booking_id -> Payment_id,
Booking_id -> Payment_method,
Booking_id -> Payment_amount

**Partial Key Dependency:** None

**Transitive Dependency:** None

**Redundancies:** None

**Anomalies:**
Insert – None.
Update – Once assigned booking_id cannot be updated.
Delete – None.

In this schema, every attribute is single-valued (scalar) making it already in 1NF. There is no partial dependency here, so it is in 2NF as well.

**2NF Redundancies:** None
Since there is no transitive dependency, it is also in 3NF. Thus, our final schema remains the same.

For a relation to be in **BCNF**,

a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super key.

Hence, this relation is in BCNF as well.

5. **Activity (Activity_id, Activity_name, Activity_charge, Activity_time, Equip_id, Trainer_id)**

**Primary Key:** Activity_id

**Foreign Key:** Equip_id, Trainer_id

**Dependencies:**
Activity_id -> Activity_name,
Activity_id -> Activity_charge
Activity_id -> Activity_time

**Partial Key Dependency:** None

**Transitive Dependency:** None

**Redundancies:** None

**Anomalies:**
Insert – None.
Update – None
Delete – None.

In this schema, every attribute is not a single-valued (scalar) making it already in 1NF. There is no partial dependency here, so it is in 2NF as well.

**2NF Redundancies:** None
Since there is no transitive dependency, it is also in 3NF. Thus, our final schema remains the same.

For a relation to be in **BCNF**,

a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super key.

Hence, this relation is in BCNF as well.

## 6. Equipment (Equip_id, Equip_name, Equip_quantity, Equip_condition, Date_of_purchase)

**Primary Key:** Equip_id

**Dependencies:**
Equip_id-> Equip_name,
Equip_id-> Equip_quantity,
Equip_id-> Equip_condition
Equip_id -> Date_of_purchase

**Partial Key Dependency:** None

**Transitive Dependency:** None

**Redundancies:** None

**Anomalies:**
Insert – None.
Update – None.
Delete – None.

In this schema, every attribute is single-valued (scalar) making it already in 1NF. There is no partial dependency here, so it is in 2NF as well.

**2NF Redundancies:** None
Since there is no transitive dependency, it is also in 3NF. Thus, our final schema remains the same.

For a relation to be in **BCNF**,

a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super key.

Hence, this relation is in BCNF as well

## 7. Trainers (Trainer_id, Trainer_name, Activity_id, activity_name)

**Primary Key:** Trainer_id

**Foreign Key:** Activity_id

**Dependencies:**
Trainer_id-> Trainer_name

**Partial Key Dependency:** None

**Transitive Dependency:** None

**Redundancies:** None

**Anomalies:**
Insert – None.
Update – None.
Delete – None.

In this, we drop activity_id and activity_name as this column is present in the Activity table and trainer_id is given to them as a foreign key.

In this schema, every attribute is single-valued (scalar) making it already in 1NF. There is no partial dependency here, so it is in 2NF as well.

**2NF Redundancies:** None
Since there is no transitive dependency, it is also in 3NF. Thus, our final schema remains the same.

For a relation to be in **BCNF**,

a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super key.

Hence, this relation is in BCNF as well

## 8. Event (Event_id, Event_name, Event_time, Event_date, Event_requirements, manager_id, Staff_id, Booking_id)

**Primary Key:** Event_id

**Foreign Key:** Manager_id, Booking_id, Staff_id

**Dependencies:**
Event_id -> Event_name,
Event_id -> Event_time,
Event_id -> Event_date,
Event_id -> Event_requirements

**Partial Key Dependency:** None

**Transitive Dependency:** None

**Redundancies:** None

**Anomalies:**
Insert – None.
Update – None.
Delete – None.

In this schema, every attribute is single-valued (scalar) making it already in 1NF. There is no partial dependency here, so it is in 2NF as well.

**2NF Redundancies:** None
Since there is no transitive dependency, it is also in 3NF. Thus, our final schema remains the same.

For a relation to be in **BCNF**,

a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super key.

Hence, this relation is in BCNF as well

## 9. Security (Security_id, Security_name, Department)

**Primary Key:** Security_id

**Dependencies:**
Security_id -> Security_name,
Security_id -> Department

**Partial Key Dependency:** None

**Transitive Dependency:** None

**Redundancies:** None

**Anomalies:**
Insert – None.
Update – None.
Delete – None.

In this schema, every attribute is single-valued (scalar) making it already in 1NF. There is no partial dependency here, so it is in 2NF as well.

**2NF Redundancies:** None
Since there is no transitive dependency, it is also in 3NF. Thus, our final schema remains the same.

For a relation to be in **BCNF**,

1. It should be in the third normal form (3NF).
2. For any dependency A → B, A must be a super key.

Hence, this relation is in BCNF as well

## 10. Staff (Staff_id, Staff_name, department)

**Primary Key:** Staff_id

**Dependencies:**
Staff_id -> Staff_name,
Staff_id -> Department

**Partial Key Dependency:** None

**Transitive Dependency:** None

**Redundancies:** None

**Anomalies:**
Insert – None.
Update – None.
Delete – None.

In this schema, every attribute is single-valued (scalar), making it already in 1NF. There is no partial dependency here, so it is in 2NF.

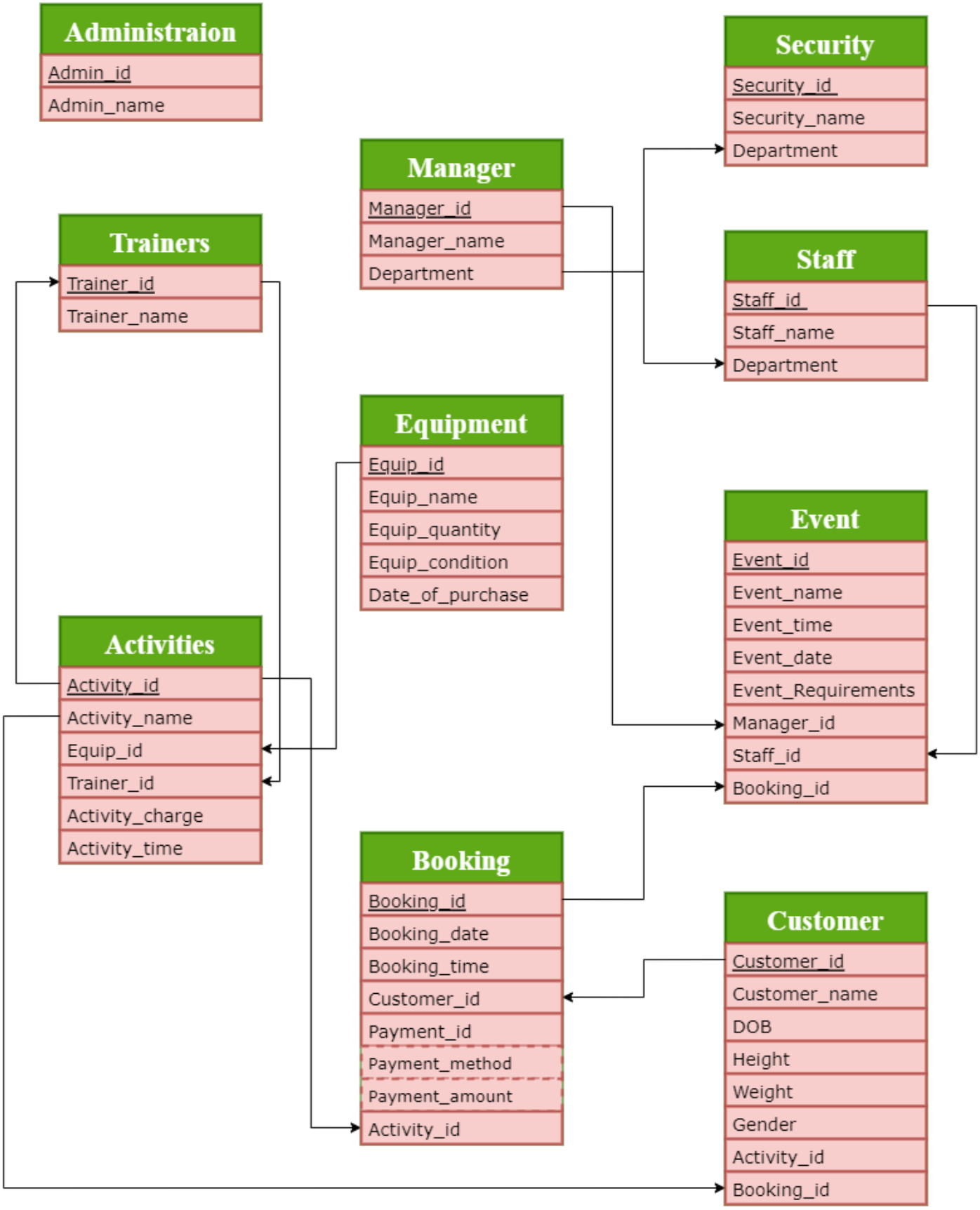
**2NF Redundancies:** None
Since there is no transitive dependency, it is also in 3NF. Thus, our final schema remains the same.

For a relation to be in **BCNF**,

1. It should be in the third normal form (3NF).
2. For any dependency A → B, A must be a super key.

Hence, this relation is in BCNF as well.

## Updated Relational Model:



# Final DDL and Select Statements

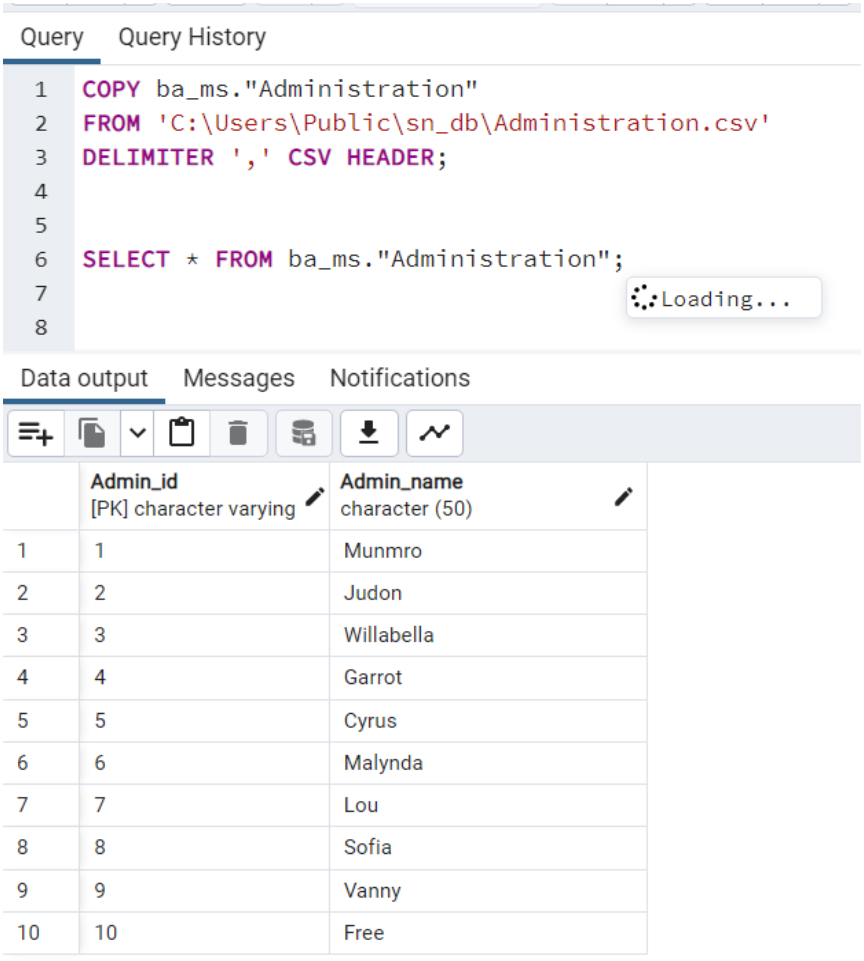
## 1. Administration (Admin_id, Admin_name)

```
---Administration

CREATE TABLE IF NOT EXISTS ba_ms."Administration"
(
    "Admin_id" character varying COLLATE pg_catalog."default" NOT NULL,
    "Admin_name" character(50) COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT "Administration_pkey" PRIMARY KEY ("Admin_id")
)
WITH (
    OIDS = FALSE
)
TABLESPACE pg_default;

ALTER TABLE IF EXISTS ba_ms."Administration"
    OWNER to postgres;

COPY ba_ms."Administration"
FROM 'C:\Users\Public\sn_db\Administration.csv'
DELIMITER ',' CSV HEADER;
```




| | Admin_id [PK] character varying | Admin_name character (50) |
|---|---|---|
| 1 | 1 | Munmro |
| 2 | 2 | Judon |
| 3 | 3 | Willabella |
| 4 | 4 | Garrot |
| 5 | 5 | Cyrus |
| 6 | 6 | Malynda |
| 7 | 7 | Lou |
| 8 | 8 | Sofia |
| 9 | 9 | Vanny |
| 10 | 10 | Free |

## 2. Manager (Manager_id, Manager_name, department)

```
--- Manager ---

CREATE TABLE IF NOT EXISTS ba_ms."Manager"
(
    "Manager_id" character varying COLLATE pg_catalog."default" NOT NULL,
    "Manager_name" character(50) COLLATE pg_catalog."default" NOT NULL,
    "Department" character(50) COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT "Manager_id" PRIMARY KEY ("Manager_id")
)
WITH (
    OIDS = FALSE
)
TABLESPACE pg_default;

ALTER TABLE IF EXISTS ba_ms."Manager"
    OWNER to postgres;


COPY ba_ms."Manager"
FROM 'C:\Users\Public\sn_db\Manager.csv'
DELIMITER ',' CSV HEADER;
```

Query Query History

```
9
10  COPY ba_ms."Manager"
11  FROM 'C:\Users\Public\sn_db\Manager.csv'
12  DELIMITER ',' CSV HEADER;
13
14
15  SELECT * FROM ba_ms."Manager";
16
```

Data output Messages Notifications

| | Manager_id [PK] character varying | Manager_name character (50) | Department character (50) |
|---|---|---|---|
| 1 | 1 | Jo | Activity |
| 2 | 2 | Clay | Equipment |
| 3 | 3 | Reggis | Staff |
| 4 | 4 | Elfrieda | Security |
| 5 | 5 | Babita | Trainer |
| 6 | 6 | Paloma | Event |
| 7 | 7 | Aurlie | Booking |
| 8 | 8 | Georgina | Activity |
| 9 | 9 | Sam | Equipment |
| 10 | 10 | Dre | Staff |
| 11 | 11 | Dirk | Security |
| 12 | 12 | Gerrilee | Trainer |
| 13 | 13 | Elmira | Event |
| 14 | 14 | Annmaria | Booking |

Total rows: 15 of 15 Query complete 00:00:00.064

## 3. Staff (Staff_id, Staff_name, Department)

```
--- Staff ---

CREATE TABLE IF NOT EXISTS ba_ms."Staff"
(
    "Staff_id" character varying COLLATE pg_catalog."default" NOT NULL,
    "Staff_name" character(100) COLLATE pg_catalog."default" NOT NULL,
    "Department" character(100) COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT "Staff_pkey" PRIMARY KEY ("Staff_id")
)
WITH (
    OIDS = FALSE
)
TABLESPACE pg_default;

ALTER TABLE IF EXISTS ba_ms."Staff"
    OWNER to postgres;

COPY ba_ms."Staff"
FROM 'C:\Users\Public\sn_db\Staff.csv'
DELIMITER ',' CSV HEADER;
```

Data output | Messages | Notifications

| | Staff_id [PK] character varying | Staff_name character (100) | Department character (100) |
|---|---|---|---|
| 1 | 1 | Karlik | Activity |
| 2 | 2 | Melosa | Equipment |
| 3 | 3 | Ame | Staff |
| 4 | 4 | Dawn | Security |
| 5 | 5 | Francesco | Trainer |
| 6 | 6 | Reyna | Event |
| 7 | 7 | Barbe | Booking |
| 8 | 8 | Ivy | Activity |
| 9 | 9 | Tamiko | Equipment |
| 10 | 10 | Wally | Staff |
| 11 | 11 | Diann | Security |
| 12 | 12 | Michael | Trainer |
| 13 | 13 | Stanleigh | Event |
| 14 | 14 | Florance | Booking |
| 15 | 15 | Claudia | Activity |
| 16 | 16 | Mimi | Equipment |
| 17 | 17 | Christoper | Staff |
| 18 | 18 | Eden | Security |
| 19 | 19 | Elianore | Trainer |
| 20 | 20 | Jere | Event |
| 21 | 21 | Samaria | Booking |

Total rows: 50 of 50 | Query complete 00:00:00.070

## 4. Security (Security_id, Security_name, Department)

```
--- Security ---

CREATE TABLE IF NOT EXISTS ba_ms."Security"
(
    "Security_id" character varying COLLATE pg_catalog."default" NOT NULL,
    "Security_name" character(100) COLLATE pg_catalog."default" NOT NULL,
    "Department" character(100) COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT "Security_pkey" PRIMARY KEY ("Security_id")
)
WITH (
    OIDS = FALSE
)
TABLESPACE pg_default;
```

```
ALTER TABLE IF EXISTS ba_ms."Security"
    OWNER to postgres;


COPY ba_ms."Security"
FROM 'C:\Users\Public\sn_db\Security.csv'
DELIMITER ',' CSV HEADER;
ACCESS
```

Query | Query History

```
27  FROM 'C:\Users\Public\sn_db\Security.csv'
28  DELIMITER ',' CSV HEADER;
29
30
31  SELECT * FROM ba_ms."Security";
```

Data output | Messages | Notifications | Loading...

| | Security_id [PK] character varying | Security_name character (100) | Department character (100) |
|---|---|---|---|
| 1 | 1 | Nataline | Activity |
| 2 | 2 | Arlena | Equipment |
| 3 | 3 | Harland | Staff |
| 4 | 4 | Nicola | Security |
| 5 | 5 | Teddie | Trainer |
| 6 | 6 | Rickert | Event |
| 7 | 7 | Marius | Booking |
| 8 | 8 | Deonne | Activity |
| 9 | 9 | Biron | Equipment |
| 10 | 10 | Agnola | Staff |
| 11 | 11 | Melva | Security |
| 12 | 12 | Brinn | Trainer |
| 13 | 13 | Marcelo | Event |
| 14 | 14 | Gwynne | Booking |
| 15 | 15 | Sherlocke | Activity |
| 16 | 16 | Felicio | Equipment |

Total rows: 50 of 50 | Query complete 00:00:00.062

## 5. Customer (Customer_id, Customer_name, DOB, age, Height, Weight, Gender, Activtiy_id, Booking_id)

```
--- Customer ---

CREATE TABLE IF NOT EXISTS ba_ms."Customer"
(
    "Customer_id" character varying COLLATE pg_catalog."default" NOT NULL,
    "Customer_name" character(100) COLLATE pg_catalog."default" NOT NULL,
    "Gender" character(100) COLLATE pg_catalog."default" NOT NULL,
    "Activity_id" character varying COLLATE pg_catalog."default" NOT NULL,
    "DOB" date NOT NULL,
    "Height" numeric NOT NULL,
    "Weight" numeric NOT NULL,
    CONSTRAINT "Customer_pkey" PRIMARY KEY ("Customer_id"),
    CONSTRAINT "Activity_id" FOREIGN KEY ("Activity_id")
        REFERENCES ba_ms."Activity" ("Activity_id") MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
)
WITH (
    OIDS = FALSE
)
TABLESPACE pg_default;

ALTER TABLE IF EXISTS ba_ms."Customer"
    OWNER to postgres;

COPY ba_ms."Customer"
FROM 'C:\Users\Public\sn_db\Customer.csv'
DELIMITER ',' CSV HEADER;
```

Query | Query History

```
72  COPY ba_ms."Customer"
73  FROM 'C:\Users\Public\sn_db\Customer.csv'
74  DELIMITER ',' CSV HEADER;
```

Data output | Messages | Notifications

| | Customer_id [PK] character varying | Customer_name character (100) | Gender character (100) | Activity_id character varying | DOB date | Height numeric | Weight numeric |
|---|---|---|---|---|---|---|---|
| 1 | 1 | Pepillo | M | 1 | 2003-07-09 | 163.07 | 84.84 |
| 2 | 2 | Elise | F | 2 | 2003-07-10 | 127.31 | 55.76 |
| 3 | 3 | Raquela | F | 3 | 2003-07-11 | 133.77 | 81.73 |
| 4 | 4 | Martino | M | 4 | 2003-07-12 | 154.11 | 101.24 |
| 5 | 5 | Emmanuel | M | 5 | 1990-12-11 | 157.57 | 75.65 |
| 6 | 6 | Bordie | M | 6 | 1990-12-12 | 104.37 | 112.1 |
| 7 | 7 | Baudoin | M | 7 | 2003-07-09 | 130.98 | 100.2 |
| 8 | 8 | Hillel | M | 8 | 1988-10-08 | 138.96 | 48.6 |
| 9 | 9 | Glenda | F | 9 | 1988-10-09 | 146.67 | 69.99 |
| 10 | 10 | Lilly | F | 10 | 1988-10-10 | 105.23 | 50.49 |
| 11 | 11 | Cal | M | 11 | 1988-10-11 | 113.79 | 91.99 |
| 12 | 12 | Octavius | M | 12 | 2000-04-08 | 119.33 | 113.92 |
| 13 | 13 | Phillipe | M | 13 | 2000-04-09 | 131.5 | 55.15 |
| 14 | 14 | Maxwell | M | 14 | 2000-04-10 | 165.64 | 53.78 |
| 15 | 15 | Gare | M | 15 | 2000-04-11 | 187.46 | 124.97 |
| 16 | 16 | Jobye | F | 1 | 2001-09-04 | 169.28 | 48.76 |
| 17 | 17 | Abbe | M | 2 | 2001-09-05 | 195.89 | 46.06 |

Total rows: 50 of 50 | Query complete 00:00:00.058

## 6. Booking (Booking_id, Booking_date, Booking_time, Customer_id, Payment_id, Payment_method, Payment_amount, Activity_id)

```
--- Booking ---

CREATE TABLE IF NOT EXISTS ba_ms."Booking"
(
    "Booking_id" character varying COLLATE pg_catalog."default" NOT NULL,
    "Booking_date" date NOT NULL,
    "Booking_time" time without time zone NOT NULL,
    "Payment_id" character varying COLLATE pg_catalog."default" NOT NULL,
    "Payment_method" character(100) COLLATE pg_catalog."default" NOT NULL,
    "Payment_amount" numeric NOT NULL,
    "Activity_id" character varying COLLATE pg_catalog."default" NOT NULL,
    "Customer_id" character varying COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT "Booking_pkey" PRIMARY KEY ("Booking_id"),
    CONSTRAINT "Activity_id" FOREIGN KEY ("Activity_id")
```

```
        REFERENCES ba_ms."Activity" ("Activity_id") MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT "Customer_id" FOREIGN KEY ("Customer_id")
        REFERENCES ba_ms."Customer" ("Customer_id") MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID
)
WITH (
    OIDS = FALSE
)
TABLESPACE pg_default;

ALTER TABLE IF EXISTS ba_ms."Booking"
    OWNER to postgres;

COPY ba_ms."Booking"
FROM 'C:\Users\Public\sn_db\Booking.csv'
DELIMITER ',' CSV HEADER;
```

Query Query History

```
34
35  COPY ba_ms."Booking"
36  FROM 'C:\Users\Public\sn_db\Booking.csv'
37  DELIMITER ',' CSV HEADER;
```

Data output Messages Notifications

| | Booking_id [PK] character varying | Booking_date date | Booking_time time without time zone | Payment_id character varying | Payment_method character (100) | Payment_amount numeric | Activity_id character varying | Customer_id character varying |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 2023-05-01 | 00:37:00 | 201465000000000 | Cash | 6073 | 1 | 1 |
| 2 | 2 | 2023-05-02 | 08:02:00 | 5100150000000... | Card | 3169 | 2 | 2 |
| 3 | 3 | 2023-05-03 | 19:36:00 | 4405820000000... | Upi | 1736 | 3 | 3 |
| 4 | 4 | 2023-05-04 | 02:01:00 | 5602230000000... | Cash | 3204 | 4 | 4 |
| 5 | 5 | 2023-05-05 | 10:31:00 | 3581300000000... | Card | 3564 | 5 | 5 |
| 6 | 6 | 2023-05-06 | 16:28:00 | 5602240000000... | Upi | 8126 | 6 | 6 |
| 7 | 7 | 2023-05-07 | 18:43:00 | 3547000000000... | Cash | 2597 | 7 | 7 |
| 8 | 8 | 2023-05-08 | 08:40:00 | 3576590000000... | Card | 9213 | 8 | 8 |
| 9 | 9 | 2023-05-09 | 23:08:00 | 3569860000000... | Upi | 3914 | 9 | 9 |
| 10 | 10 | 2023-05-10 | 17:53:00 | 3533690000000... | Cash | 2443 | 10 | 10 |
| 11 | 11 | 2023-05-11 | 19:06:00 | 30571100000000 | Card | 3360 | 11 | 11 |
| 12 | 12 | 2023-05-12 | 10:44:00 | 5610670000000... | Upi | 6068 | 12 | 12 |
| 13 | 13 | 2023-05-13 | 05:54:00 | 3551460000000... | Cash | 5812 | 13 | 13 |
| 14 | 14 | 2023-05-14 | 19:15:00 | 5007670000000... | Card | 7198 | 14 | 14 |
| 15 | 15 | 2023-05-15 | 08:32:00 | 3567710000000... | Upi | 7780 | 15 | 15 |
| 16 | 16 | 2023-05-16 | 10:02:00 | 3550750000000... | Cash | 9715 | 1 | 16 |
| 17 | 17 | 2023-05-17 | 17:33:00 | 5602220000000... | Card | 2676 | 2 | |

Table truncated

Total rows: 50 of 50 Query complete 00:00:00.073

## 7. Event (Event_id, Event_name, Event_time, Event_date, Event_requirements, manager_id, Staff_id, Booking_id)

```
--- Event ---

CREATE TABLE IF NOT EXISTS ba_ms."Event"
(
    "Event_id" character varying COLLATE pg_catalog."default" NOT NULL,
    "Event_name" character(100) COLLATE pg_catalog."default" NOT NULL,
    "Event_time" time without time zone NOT NULL,
    "Event_date" date NOT NULL,
    "Event_requirements" character varying COLLATE pg_catalog."default" NOT
NULL,
    "Manager_id" character varying COLLATE pg_catalog."default" NOT NULL,
    "Staff_id" character varying COLLATE pg_catalog."default" NOT NULL,
    "Booking_id" character varying COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT "Event_pkey" PRIMARY KEY ("Event_id"),
    CONSTRAINT "Booking_id" FOREIGN KEY ("Booking_id")
        REFERENCES ba_ms."Booking" ("Booking_id") MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION,
    CONSTRAINT "Manager_id" FOREIGN KEY ("Manager_id")
        REFERENCES ba_ms."Manager" ("Manager_id") MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION,
    CONSTRAINT "Staff_id" FOREIGN KEY ("Staff_id")
        REFERENCES ba_ms."Staff" ("Staff_id") MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
)
WITH (
    OIDS = FALSE
)
TABLESPACE pg_default;

ALTER TABLE IF EXISTS ba_ms."Event"
    OWNER to postgres;

COPY ba_ms."Event"
FROM 'C:\Users\Public\sn_db\Event.csv'
DELIMITER ',' CSV HEADER;
```

Query  Query History

```
79
80  COPY ba_ms."Event"
81  FROM 'C:\Users\Public\sn_db\Event.csv'
82  DELIMITER ',' CSV HEADER;
```

Loading...

Data output  Messages  Notifications

| | Event_id [PK] character varying | Event_name character (100) | Event_time time without time zone | Event_date date | Event_requirements character varying | Manager_id character varying | Staff_id character varying | Booking_id character varying |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | Birthday | 00:29:00 | 2023-06-02 | baloons | 1 | 1 | 1 |
| 2 | 2 | Wedding | 02:40:00 | 2024-03-11 | lights | 2 | 2 | 2 |
| 3 | 3 | Office Party | 12:52:00 | 2023-11-09 | cardboards | 3 | 3 | 3 |
| 4 | 4 | Marriage Party | 20:05:00 | 2024-10-01 | cartoons | 4 | 4 | 4 |
| 5 | 5 | Anniversary | 18:41:00 | 2023-11-11 | chairs | 5 | 5 | 5 |
| 6 | 6 | Fest | 16:14:00 | 2023-11-12 | tables | 6 | 6 | 6 |
| 7 | 7 | College Events | 13:57:00 | 2023-11-13 | stage | 7 | 7 | 7 |
| 8 | 8 | Companies Eve... | 06:23:00 | 2023-11-14 | photographer | 8 | 8 | 8 |
| 9 | 9 | Birthday | 15:38:00 | 2023-11-15 | cook | 9 | 9 | 9 |
| 10 | 10 | Wedding | 19:16:00 | 2023-01-05 | candles | 10 | 10 | 10 |
| 11 | 11 | Office Party | 05:48:00 | 2023-01-06 | baloons | 11 | 11 | 11 |
| 12 | 12 | Marriage Party | 19:22:00 | 2023-01-06 | lights | 12 | 12 | 12 |
| 13 | 13 | Anniversary | 20:43:00 | 2023-01-07 | cardboards | 13 | 13 | 13 |
| 14 | 14 | Fest | 07:33:00 | 2024-04-01 | cartoons | 14 | 14 | 14 |
| 15 | 15 | College Events | 21:23:00 | 2024-04-02 | chairs | 15 | 15 | 15 |
| 16 | 16 | Companies Eve... | 05:19:00 | 2024-04-03 | tables | 1 | 16 | 16 |
| 17 | 17 | Birthday | 07:16:00 | 2023-08-07 | stage | 2 | 17 | 17 |

Total rows: 50 of 50  Query complete 00:00:00.076

## 8. Trainers (Trainer_id, Trainer_name, Activity_id, activity_name)

```
--- Trainers —

CREATE TABLE IF NOT EXISTS ba_ms."Trainers"
(
    "Trainer_id" character varying COLLATE pg_catalog."default" NOT NULL,
    "Trainer_name" character(100) COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT "Trainers_pkey" PRIMARY KEY ("Trainer_id")
)
WITH (
    OIDS = FALSE
)
TABLESPACE pg_default;

ALTER TABLE IF EXISTS ba_ms."Trainers"
    OWNER to postgres;

COPY ba_ms."Trainers"
FROM 'C:\Users\Public\sn_db\Trainers.csv'
DELIMITER ',' CSV HEADER;
```

Query | Query History

```
54  COPY ba_ms."Trainers"
55  FROM 'C:\Users\Public\sn_db\Trainers.csv'
56  DELIMITER ',' CSV HEADER;
57
58
59  SELECT * FROM ba_ms."Trainers";
```

Data output | Messages | Notifications

| | Trainer_id [PK] character varying | Trainer_name character (100) |
|---|---|---|
| 1 | 1 | Merline |
| 2 | 2 | Aube |
| 3 | 3 | Ursala |
| 4 | 4 | Iolanthe |
| 5 | 5 | Crystie |
| 6 | 6 | Rooney |
| 7 | 7 | Shena |
| 8 | 8 | Justis |
| 9 | 9 | Cammie |
| 10 | 10 | Wynn |
| 11 | 11 | Katalin |
| 12 | 12 | Osborne |
| 13 | 13 | Eachelle |
| 14 | 14 | Delmer |
| 15 | 15 | Emmalynn |

Total rows: 15 of 15 | Query complete 00:00:00.066

## 9. Activity (Activity_id, Activity_name, Activity_charge, Activity_time, Equip_id, Trainer_id)

--- Activity ---

```
CREATE TABLE IF NOT EXISTS ba_ms."Activity"
(
    "Activity_id" character varying COLLATE pg_catalog."default" NOT NULL,
    "Activity_name" character(100) COLLATE pg_catalog."default" NOT NULL,
    "Activity_charge(in $)" integer NOT NULL,
    "Activity_time(in min)" integer NOT NULL,
    "Equip_id" character varying COLLATE pg_catalog."default" NOT NULL,
    "Trainer_id" character varying COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT "Activity_pkey" PRIMARY KEY ("Activity_id"),
    CONSTRAINT "Equip_id" FOREIGN KEY ("Equip_id")
```

```
        REFERENCES ba_ms."Equipment" ("Equipment_id") MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT "Trainer_id" FOREIGN KEY ("Trainer_id")
        REFERENCES ba_ms."Trainers" ("Trainer_id") MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID
)
WITH (
    OIDS = FALSE
)
TABLESPACE pg_default;

ALTER TABLE IF EXISTS ba_ms."Activity"
    OWNER to postgres;

COPY ba_ms."Activity"
FROM 'C:\Users\Public\sn_db\Activity.csv'
DELIMITER ',' CSV HEADER;
```

Query | Query History

```
63  COPY ba_ms."Activity"
64  FROM 'C:\Users\Public\sn_db\Activity.csv'
65  DELIMITER ',' CSV HEADER;
66
67
68  SELECT * FROM ba_ms."Activity";
```

Data output | Messages | Notifications

| | Activity_id [PK] character varying | Activity_name character (100) | Activity_charge(in $) integer | Activity_time(in min) integer | Equip_id character varying | Trainer_id character varying |
|---|---|---|---|---|---|---|
| 1 | 1 | Scuba Diving | 17 | 30 | 1 | 1 |
| 2 | 2 | Snorkeling | 12 | 35 | 2 | 2 |
| 3 | 3 | Paragliding | 10 | 20 | 3 | 3 |
| 4 | 4 | Parasailing | 15 | 25 | 4 | 4 |
| 5 | 5 | Flyboarding | 18 | 40 | 5 | 5 |
| 6 | 6 | Windsurfing | 16 | 45 | 6 | 6 |
| 7 | 7 | Beach Volleyball | 14 | 50 | 7 | 7 |
| 8 | 8 | Jet Ski | 9 | 60 | 8 | 8 |
| 9 | 9 | Kite surfing | 15 | 120 | 9 | 9 |
| 10 | 10 | Beach Sprint ro... | 6 | 90 | 10 | 10 |
| 11 | 11 | Banana Ride | 16 | 30 | 11 | 11 |
| 12 | 12 | Speed Boat | 15 | 45 | 12 | 12 |
| 13 | 13 | Dolphin Explora... | 17 | 60 | 13 | 13 |
| 14 | 14 | FootVolley | 12 | 25 | 14 | 14 |
| 15 | 15 | BeachCamping | 18 | 40 | 15 | 15 |

Total rows: 15 of 15 | Query complete 00:00:00.054

## 10. Equipment (Equip_id, Equip_name, Equip_quantity, Equip_condition, Date_of_purchase)

```
--- Equipment ---

CREATE TABLE IF NOT EXISTS ba_ms."Equipment"
(
    "Equipment_id" character varying COLLATE pg_catalog."default" NOT NULL,
    "Equipment_name" character(100) COLLATE pg_catalog."default" NOT NULL,
    "Equipment_quantity" numeric NOT NULL,
    "Equipment_condition" character(100) COLLATE pg_catalog."default" NOT NULL,
    "Date_of_purchase" date NOT NULL,
    CONSTRAINT "Equipment_pkey" PRIMARY KEY ("Equipment_id")
)
WITH (
    OIDS = FALSE
)
```

TABLESPACE pg_default;

ALTER TABLE IF EXISTS ba_ms."Equipment"
    OWNER to postgres;

COPY ba_ms."Equipment"
FROM 'C:\Users\Public\sn_db\Equipment.csv'
DELIMITER ',' CSV HEADER;

Query | Query History

```
44  COPY ba_ms."Equipment"
45  FROM 'C:\Users\Public\sn_db\Equipment.csv'
46  DELIMITER ',' CSV HEADER;
47
48
49  SELECT * FROM ba_ms."Equipment";
```

Data output | Messages | Notifications

| | Equipment_id [PK] character varying | Equipment_name character (100) | Equipment_quantity numeric | Equipment_condition character (100) | Date_of_purchase date |
|---|---|---|---|---|---|
| 1 | 1 | Life Jacket | 66 | new | 2018-01-09 |
| 2 | 2 | Helmet | 54 | old | 2020-02-09 |
| 3 | 3 | Boat | 26 | need to change | 2017-03-09 |
| 4 | 4 | Knives | 60 | new | 2020-04-09 |
| 5 | 5 | Hinges | 16 | old | 2020-05-09 |
| 6 | 6 | Ropes | 31 | need to change | 2019-06-09 |
| 7 | 7 | Parachutes | 20 | new | 2020-07-09 |
| 8 | 8 | Knee Gaurds | 13 | old | 2020-08-09 |
| 9 | 9 | Albow Gaurds | 70 | need to change | 2020-09-09 |
| 10 | 10 | Wistles | 76 | new | 2020-10-09 |
| 11 | 11 | Speed Boat | 52 | old | 2020-11-09 |
| 12 | 12 | Rowing Boats | 33 | need to change | 2020-12-09 |
| 13 | 13 | First Aid Kits | 23 | new | 2018-01-19 |
| 14 | 14 | Harness | 39 | old | 2020-02-02 |
| 15 | 15 | Lifebuoys | 10 | need to change | 2017-03-05 |

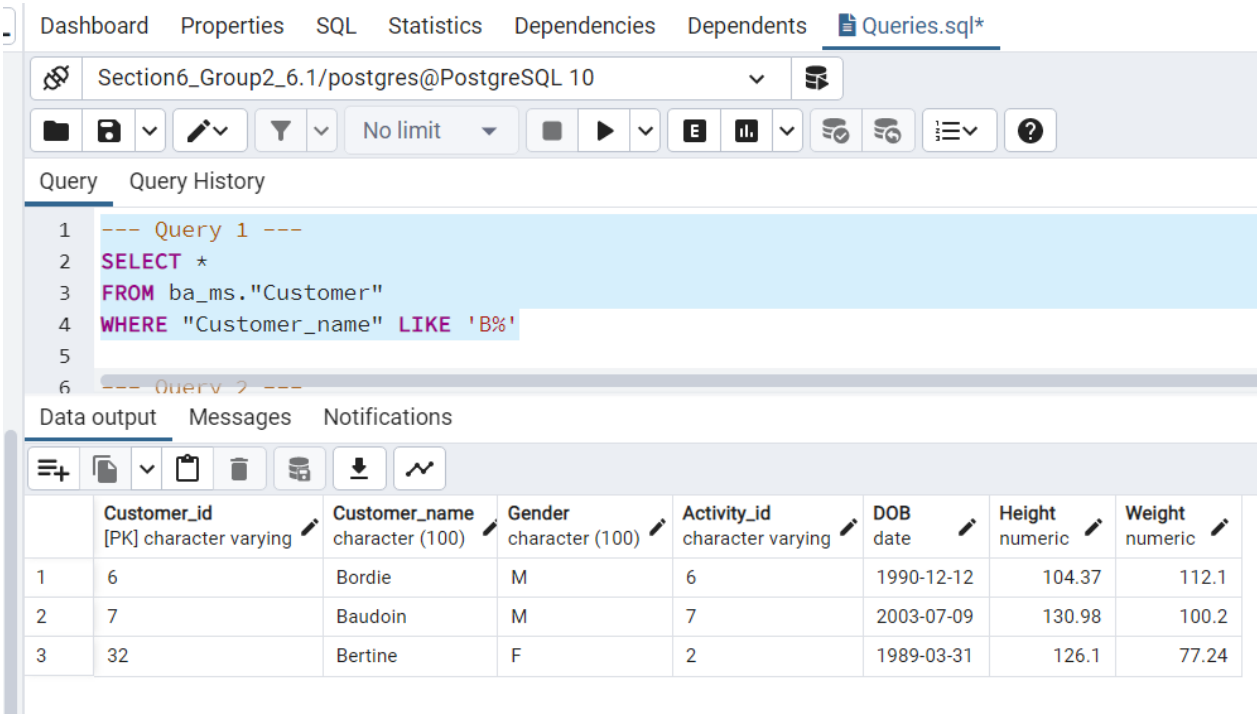
Total rows: 30 of 30 | Query complete 00:00:00.062

# SQL Queries

## Q1: To find all customer details whose name starts with ‘B’.

## Ans:

```
--- Query 1 ---
SELECT *
FROM ba_ms."Customer"
WHERE "Customer_name" LIKE 'B%'
```

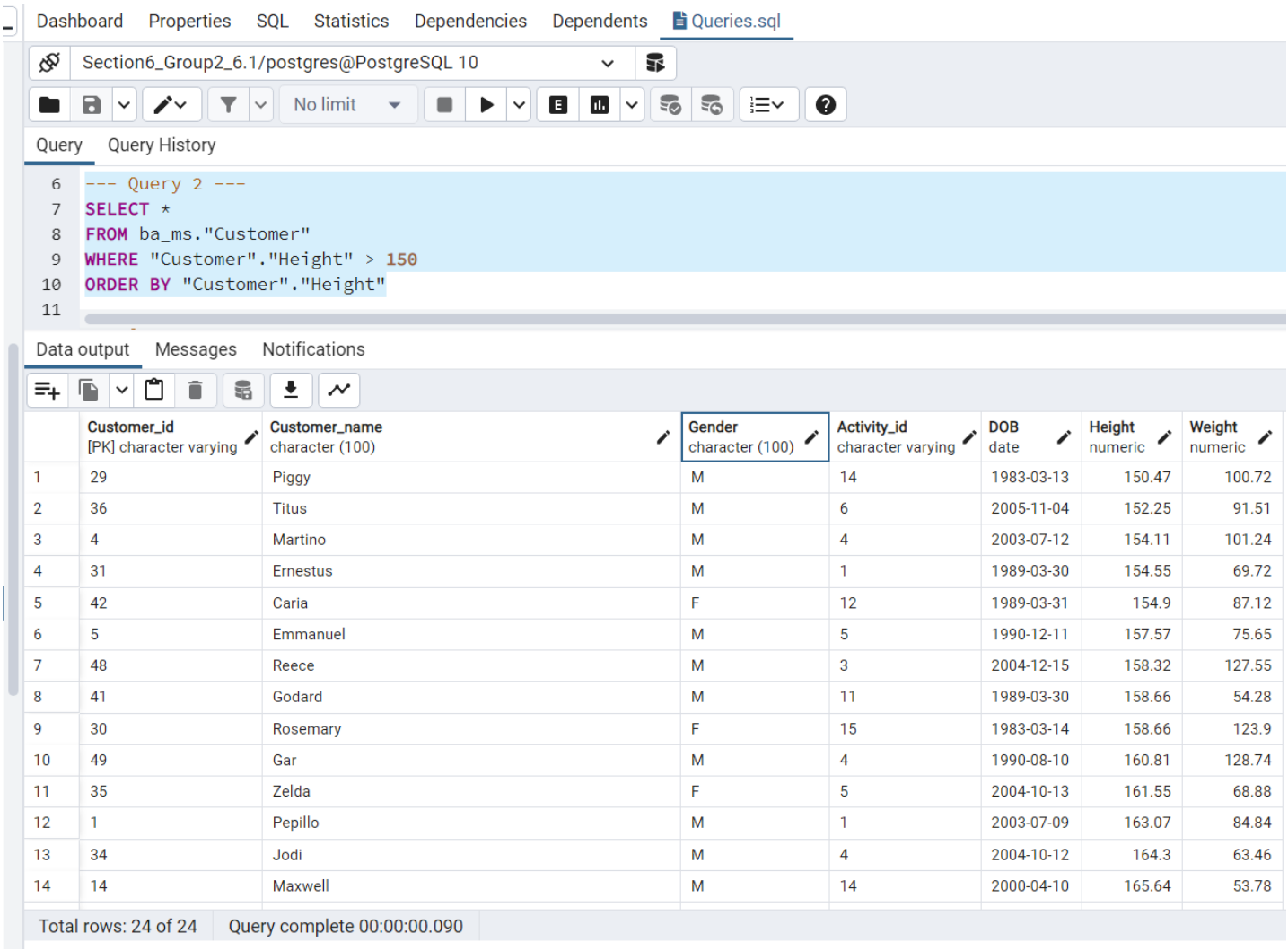



| | Customer_id [PK] character varying | Customer_name character (100) | Gender character (100) | Activity_id character varying | DOB date | Height numeric | Weight numeric |
|---|---|---|---|---|---|---|---|
| 1 | 6 | Bordie | M | 6 | 1990-12-12 | 104.37 | 112.1 |
| 2 | 7 | Baudoin | M | 7 | 2003-07-09 | 130.98 | 100.2 |
| 3 | 32 | Bertine | F | 2 | 1989-03-31 | 126.1 | 77.24 |

## Q2: To find all customers whose height is greater than 150 cm.

## Ans:

```
--- Query 2 ---
SELECT *
FROM ba_ms."Customer"
WHERE "Customer"."Height" > 150
ORDER BY "Customer"."Height"
```



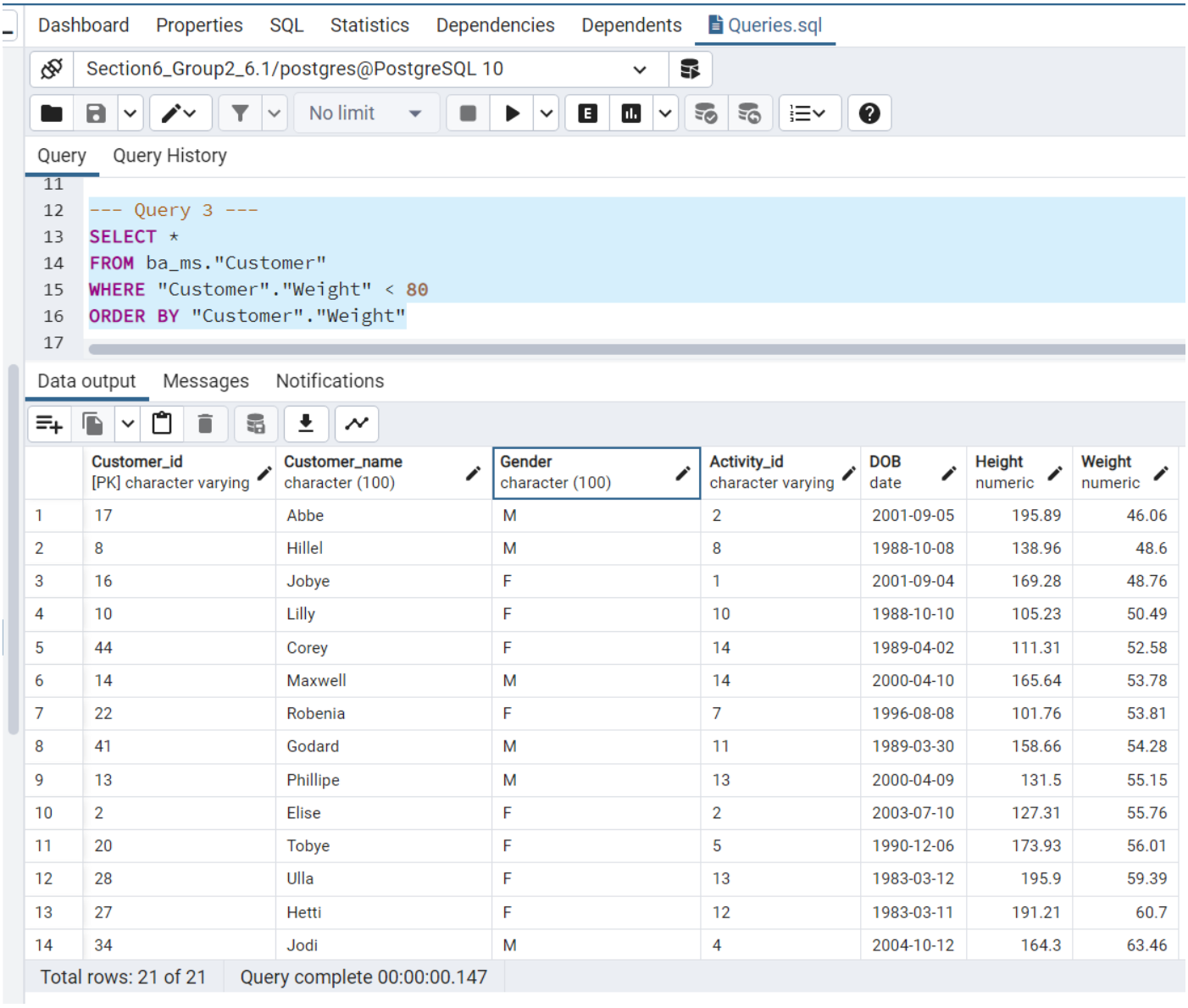

| | Customer_id [PK] character varying | Customer_name character (100) | Gender character (100) | Activity_id character varying | DOB date | Height numeric | Weight numeric |
|---|---|---|---|---|---|---|---|
| 1 | 29 | Piggy | M | 14 | 1983-03-13 | 150.47 | 100.72 |
| 2 | 36 | Titus | M | 6 | 2005-11-04 | 152.25 | 91.51 |
| 3 | 4 | Martino | M | 4 | 2003-07-12 | 154.11 | 101.24 |
| 4 | 31 | Ernestus | M | 1 | 1989-03-30 | 154.55 | 69.72 |
| 5 | 42 | Caria | F | 12 | 1989-03-31 | 154.9 | 87.12 |
| 6 | 5 | Emmanuel | M | 5 | 1990-12-11 | 157.57 | 75.65 |
| 7 | 48 | Reece | M | 3 | 2004-12-15 | 158.32 | 127.55 |
| 8 | 41 | Godard | M | 11 | 1989-03-30 | 158.66 | 54.28 |
| 9 | 30 | Rosemary | F | 15 | 1983-03-14 | 158.66 | 123.9 |
| 10 | 49 | Gar | M | 4 | 1990-08-10 | 160.81 | 128.74 |
| 11 | 35 | Zelda | F | 5 | 2004-10-13 | 161.55 | 68.88 |
| 12 | 1 | Pepillo | M | 1 | 2003-07-09 | 163.07 | 84.84 |
| 13 | 34 | Jodi | M | 4 | 2004-10-12 | 164.3 | 63.46 |
| 14 | 14 | Maxwell | M | 14 | 2000-04-10 | 165.64 | 53.78 |

Total rows: 24 of 24 Query complete 00:00:00.090

## Q3: To find all customers whose weight is less than 80 Kg.

## Ans:

```
--- Query 3 ---
SELECT *
FROM ba_ms."Customer"
WHERE "Customer"."Weight" < 80
ORDER BY "Customer"."Weight"
```

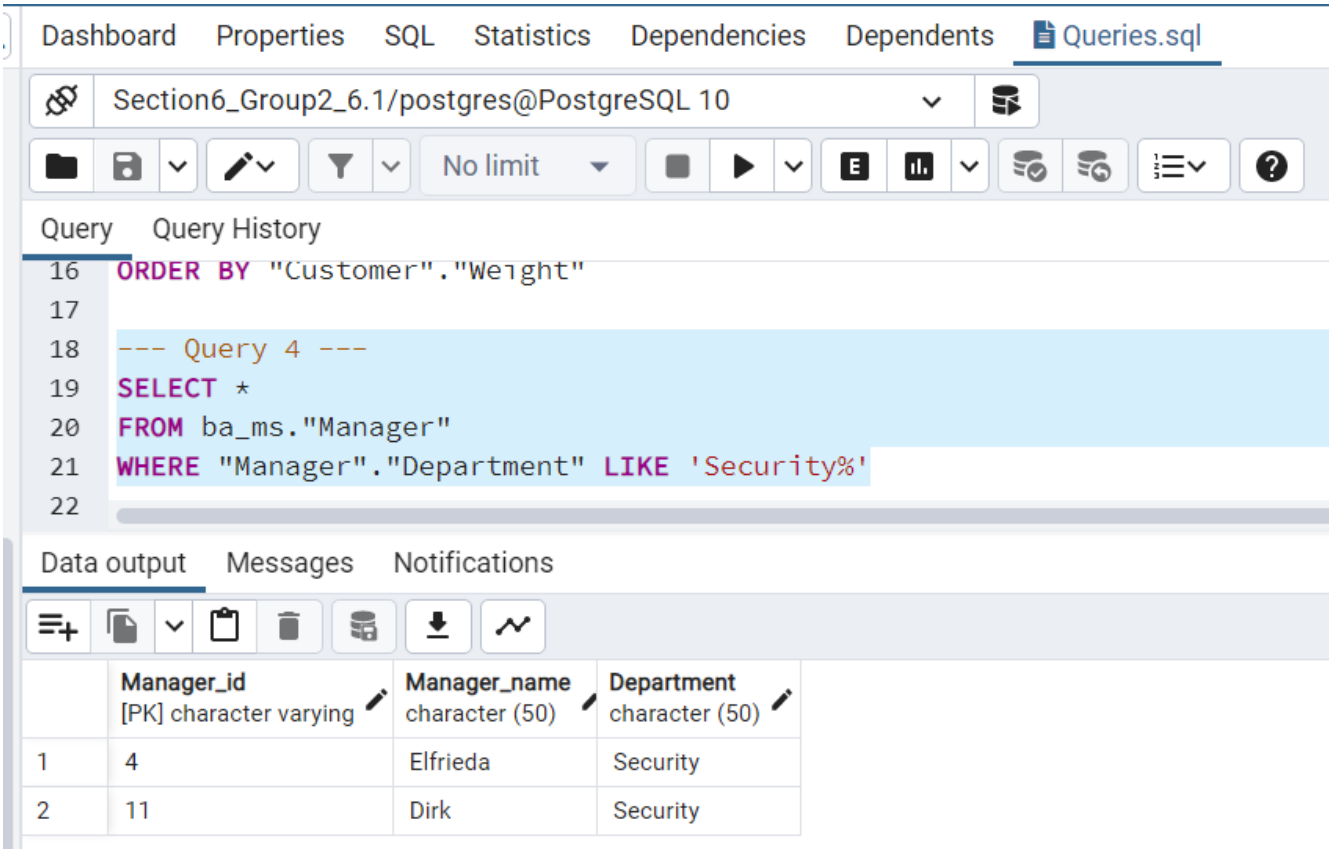



| | Customer_id [PK] character varying | Customer_name character (100) | Gender character (100) | Activity_id character varying | DOB date | Height numeric | Weight numeric |
|---|---|---|---|---|---|---|---|
| 1 | 17 | Abbe | M | 2 | 2001-09-05 | 195.89 | 46.06 |
| 2 | 8 | Hillel | M | 8 | 1988-10-08 | 138.96 | 48.6 |
| 3 | 16 | Jobye | F | 1 | 2001-09-04 | 169.28 | 48.76 |
| 4 | 10 | Lilly | F | 10 | 1988-10-10 | 105.23 | 50.49 |
| 5 | 44 | Corey | F | 14 | 1989-04-02 | 111.31 | 52.58 |
| 6 | 14 | Maxwell | M | 14 | 2000-04-10 | 165.64 | 53.78 |
| 7 | 22 | Robenia | F | 7 | 1996-08-08 | 101.76 | 53.81 |
| 8 | 41 | Godard | M | 11 | 1989-03-30 | 158.66 | 54.28 |
| 9 | 13 | Phillipe | M | 13 | 2000-04-09 | 131.5 | 55.15 |
| 10 | 2 | Elise | F | 2 | 2003-07-10 | 127.31 | 55.76 |
| 11 | 20 | Tobye | F | 5 | 1990-12-06 | 173.93 | 56.01 |
| 12 | 28 | Ulla | F | 13 | 1983-03-12 | 195.9 | 59.39 |
| 13 | 27 | Hetti | F | 12 | 1983-03-11 | 191.21 | 60.7 |
| 14 | 34 | Jodi | M | 4 | 2004-10-12 | 164.3 | 63.46 |

**Q4: To find all managers who work in the security department.**

**Ans:**

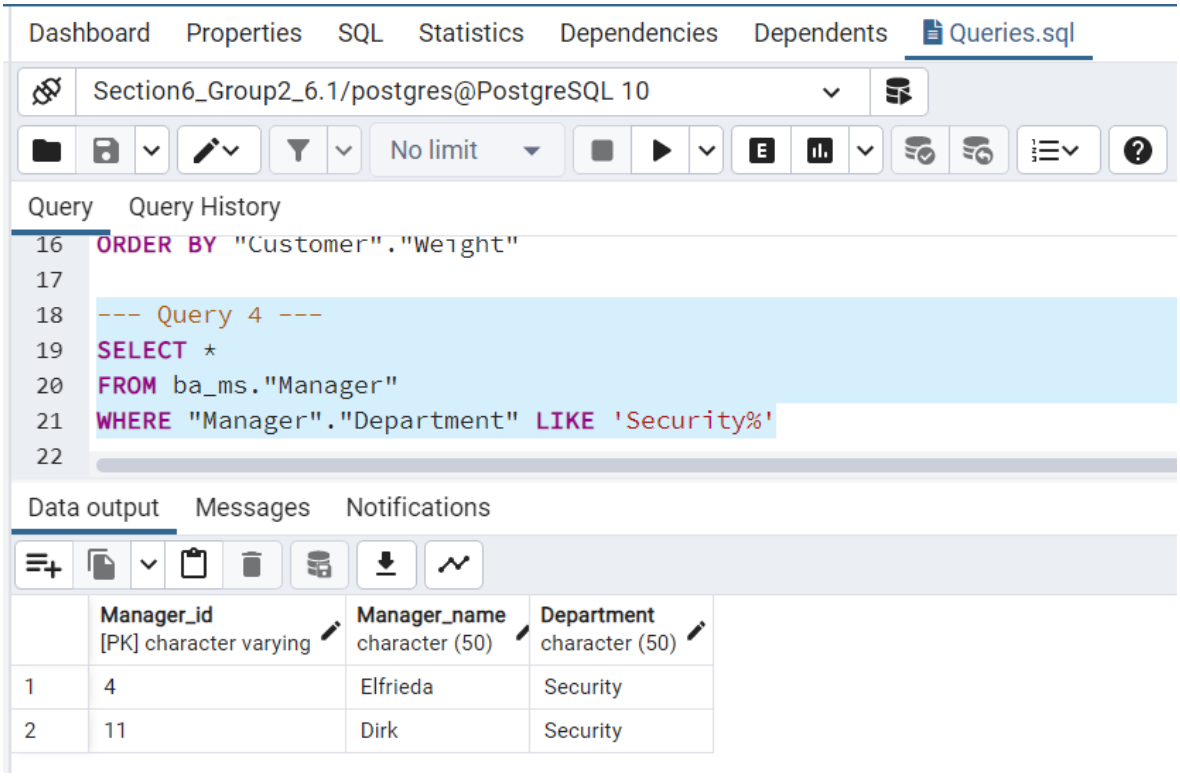
```
--- Query 4 ---
SELECT *
FROM ba_ms."Manager"
WHERE "Manager"."Department" LIKE 'Security%'
```



## Q5: Display Equipment_id and Equipment_name whose condition is old.

## Ans:

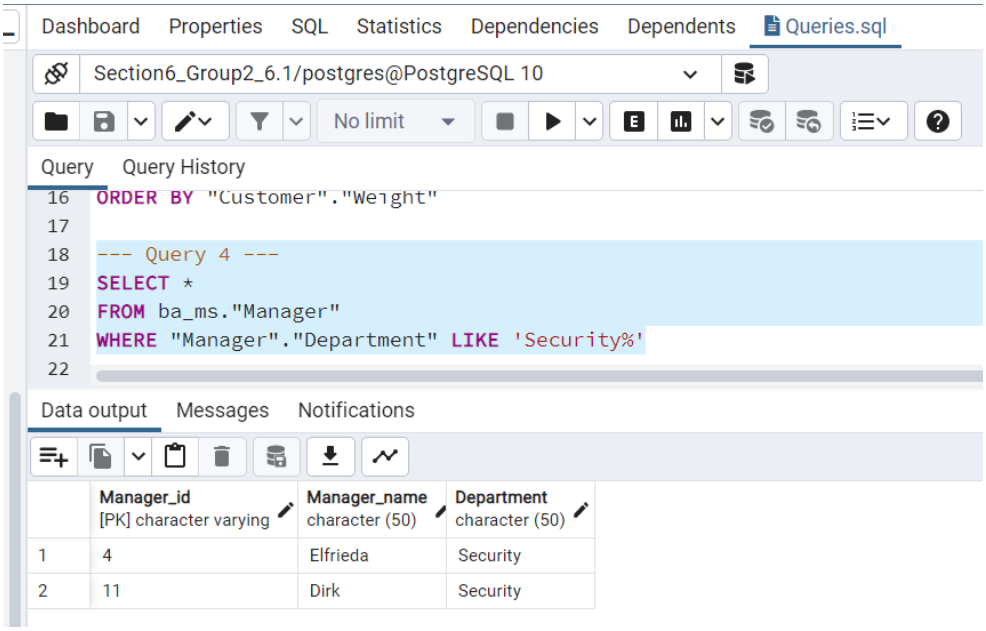
--- Query 5 ---
SELECT ba_ms."Equipment"."Equipment_id", ba_ms."Equipment"."Equipment_name"
FROM ba_ms."Equipment"
WHERE "Equipment"."Equipment_condition" LIKE 'old%'



## Q6: Display all the activity names corresponding to all the bookings made.

## Ans:

```
--- Query 6 ---
SELECT ba_ms."Booking"."Booking_id", ba_ms."Booking"."Activity_id",
ba_ms."Activity"."Activity_name"
FROM ba_ms."Activity"
INNER JOIN ba_ms."Booking" ON ba_ms."Booking"."Activity_id" =
ba_ms."Activity"."Activity_id"
```

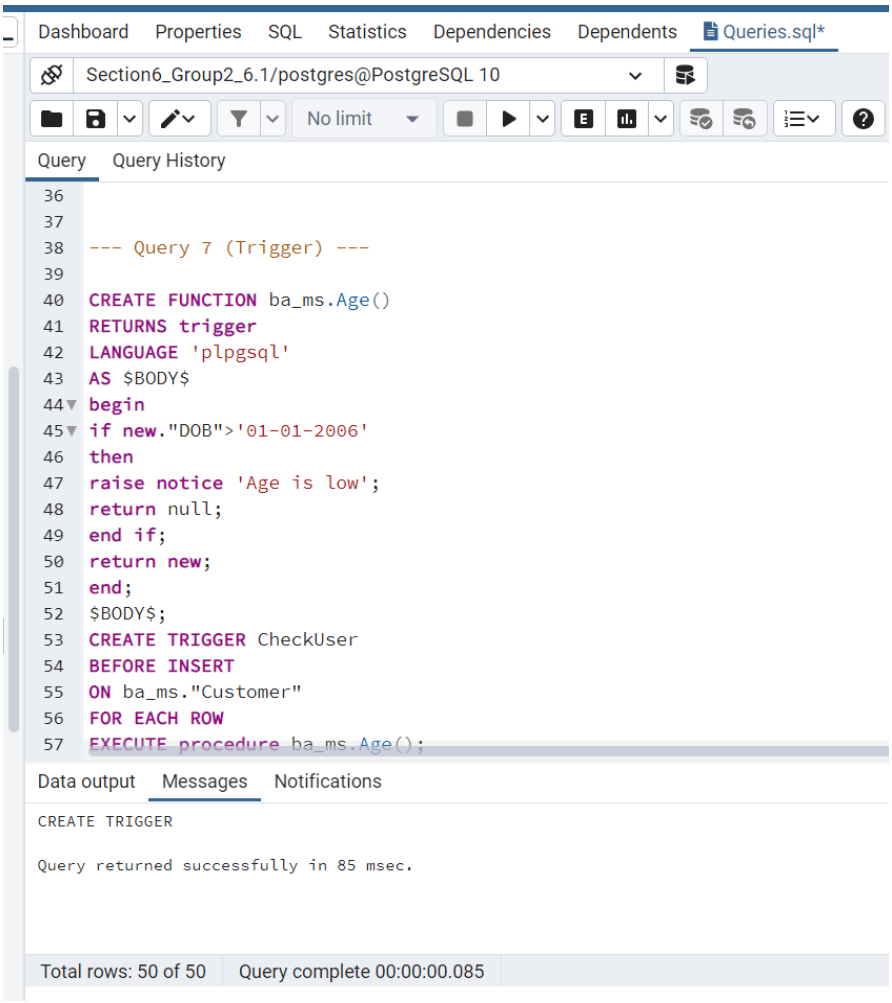


**Q7: <u>Trigger:</u> If any customer is below the required age then they can't insert their data in customer details.**

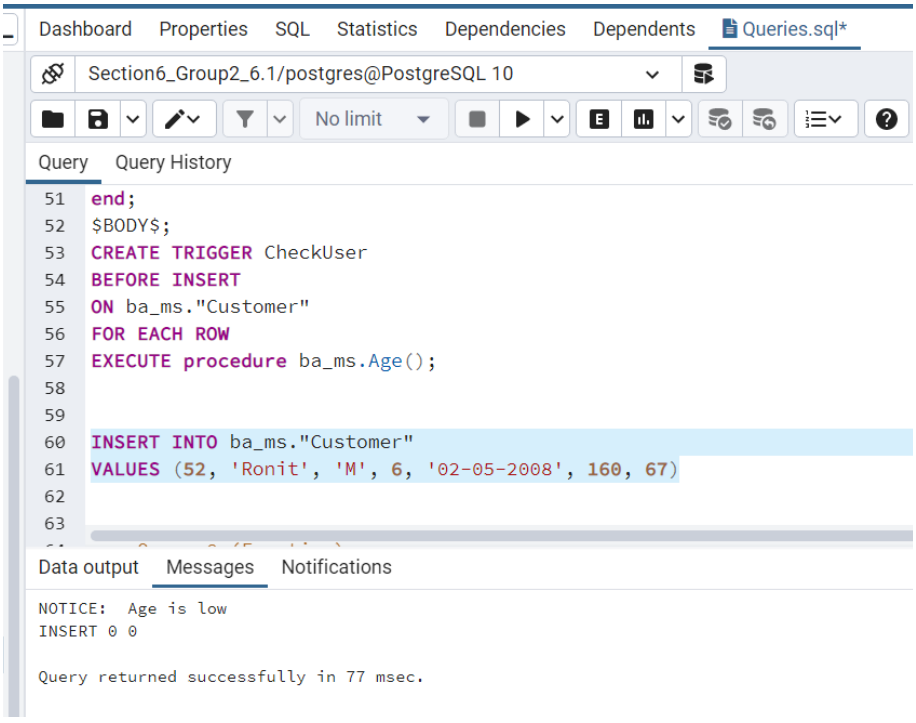
**Ans:**

--- Query 7 (Trigger) ---

```
CREATE FUNCTION ba_ms.Age()
RETURNS trigger
LANGUAGE 'plpgsql'
AS $BODY$
begin
if new."DOB">'01-01-2006'
then
raise notice 'Age is low';
return null;
end if;
return new;
end;
$BODY$;
CREATE TRIGGER CheckUser
BEFORE INSERT
ON ba_ms."Customer"
FOR EACH ROW
EXECUTE procedure ba_ms.Age();
```



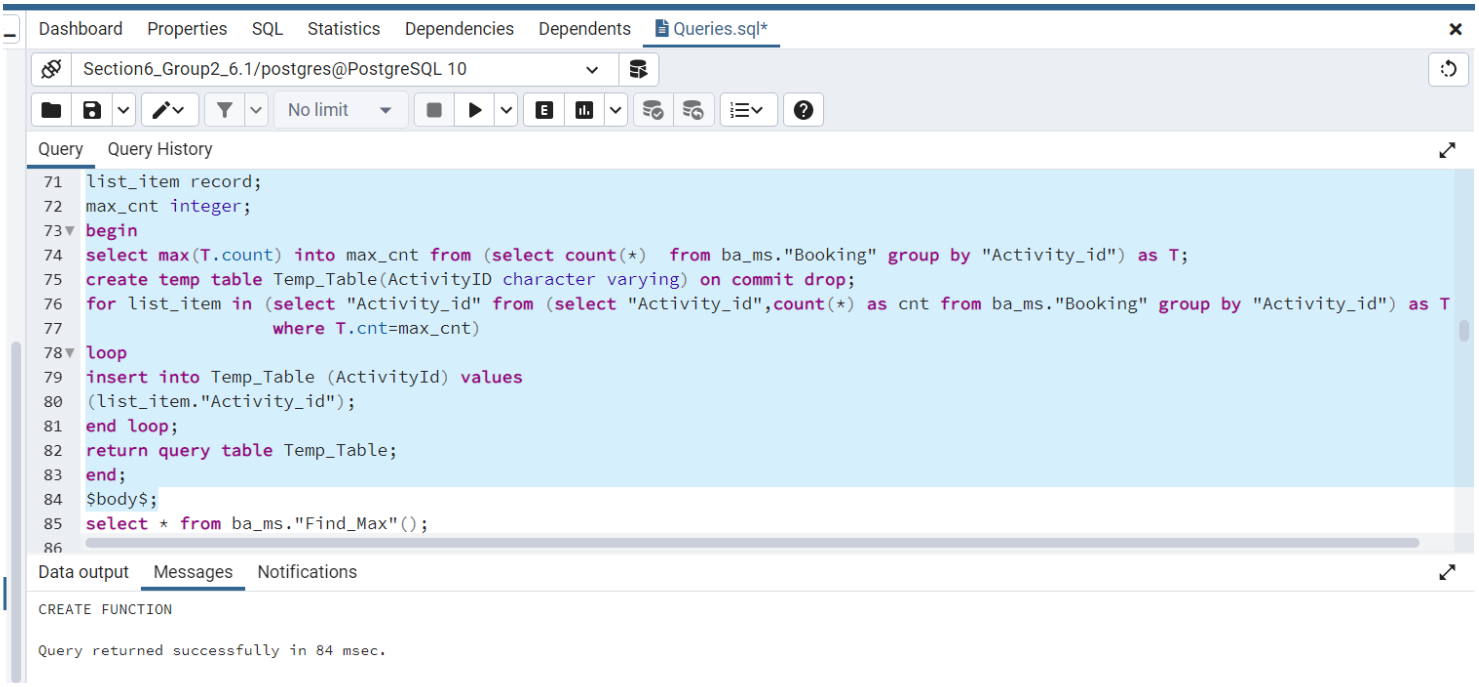

INSERT INTO ba_ms."Customer"
VALUES (52, 'Ronit', 'M', 6, '02-05-2008', 160, 67)



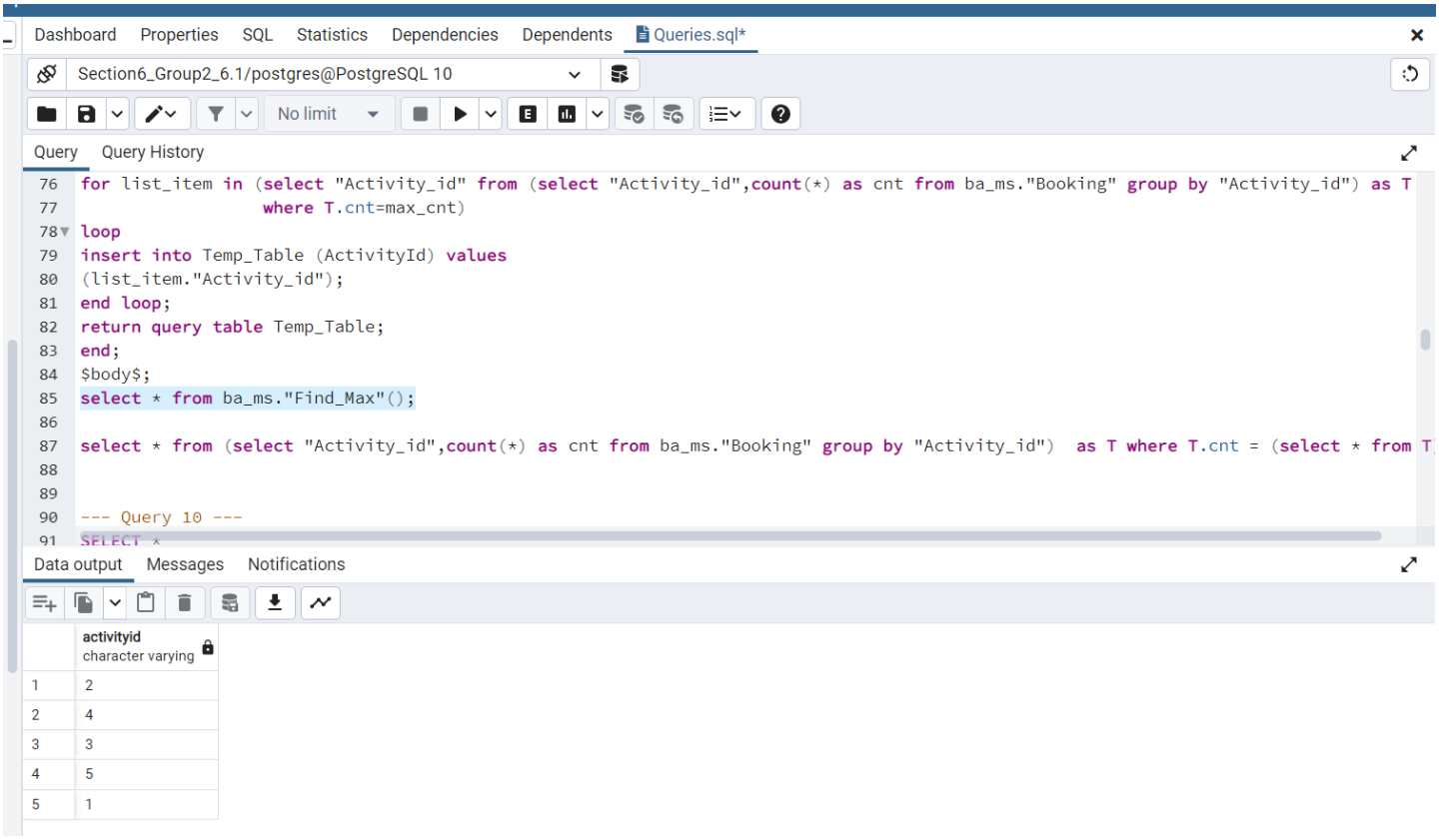
## Q8: Function: In our function return the activity id in which the number of bookings is highest.

## Ans:

```
create or replace function ba_ms."Find_Max"()
returns table(ActivityID character varying)
language 'plpgsql'
as $body$
declare
list_item record;
max_cnt integer;
begin
select max(T.count) into max_cnt from (select count(*)  from ba_ms."Booking" group by
"Activity_id") as T;
create temp table Temp_Table(ActivityID character varying) on commit drop;
for list_item in (select "Activity_id" from (select "Activity_id",count(*) as cnt from
ba_ms."Booking" group by "Activity_id") as T
                                    where T.cnt=max_cnt)
loop
insert into Temp_Table (ActivityId) values
(list_item."Activity_id");
end loop;
return query table Temp_Table;
end;
$body$;
```



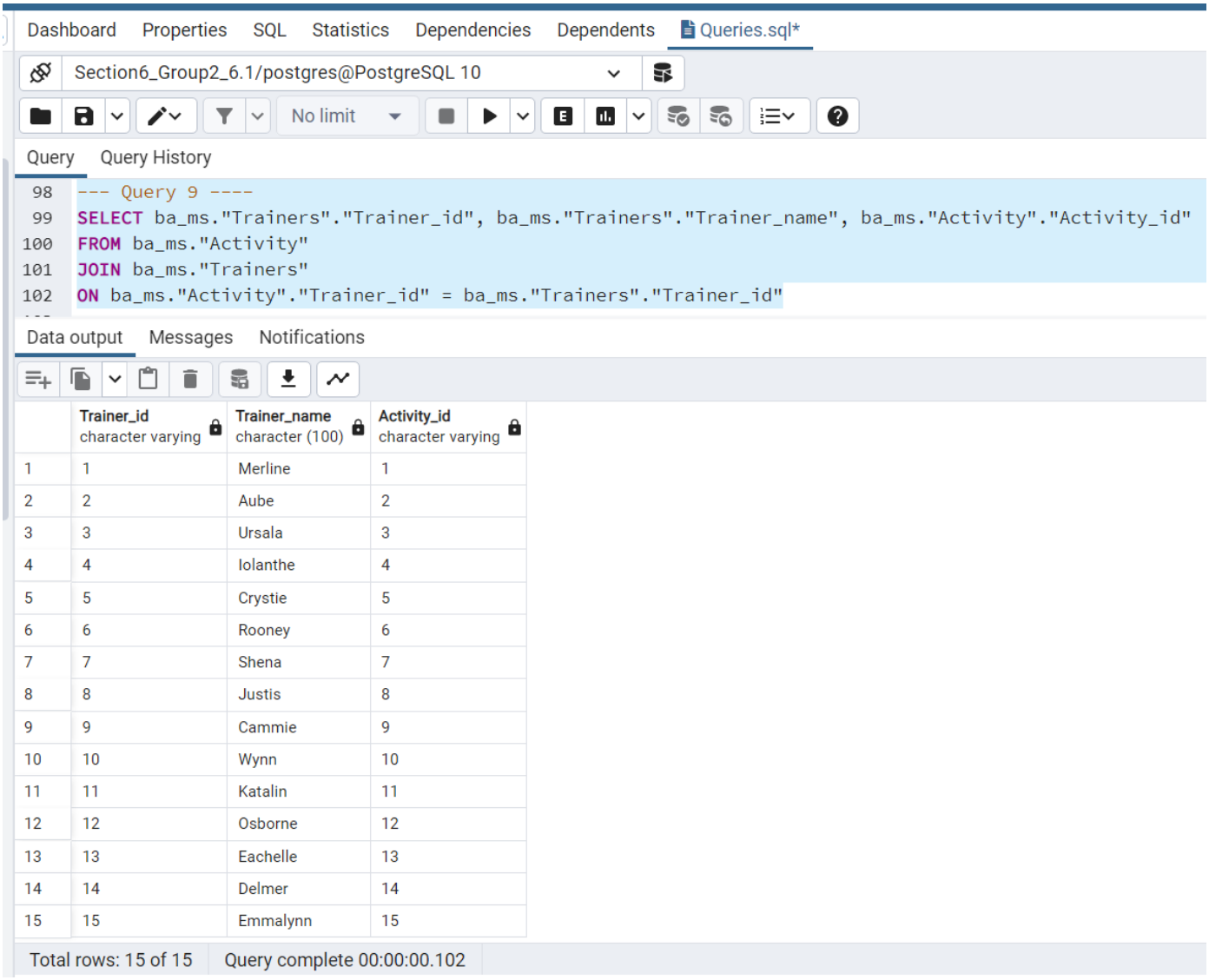
select * from ba_ms."Find_Max"();



## Q9: Print names of trainers associated with corresponding activity.

## Ans:

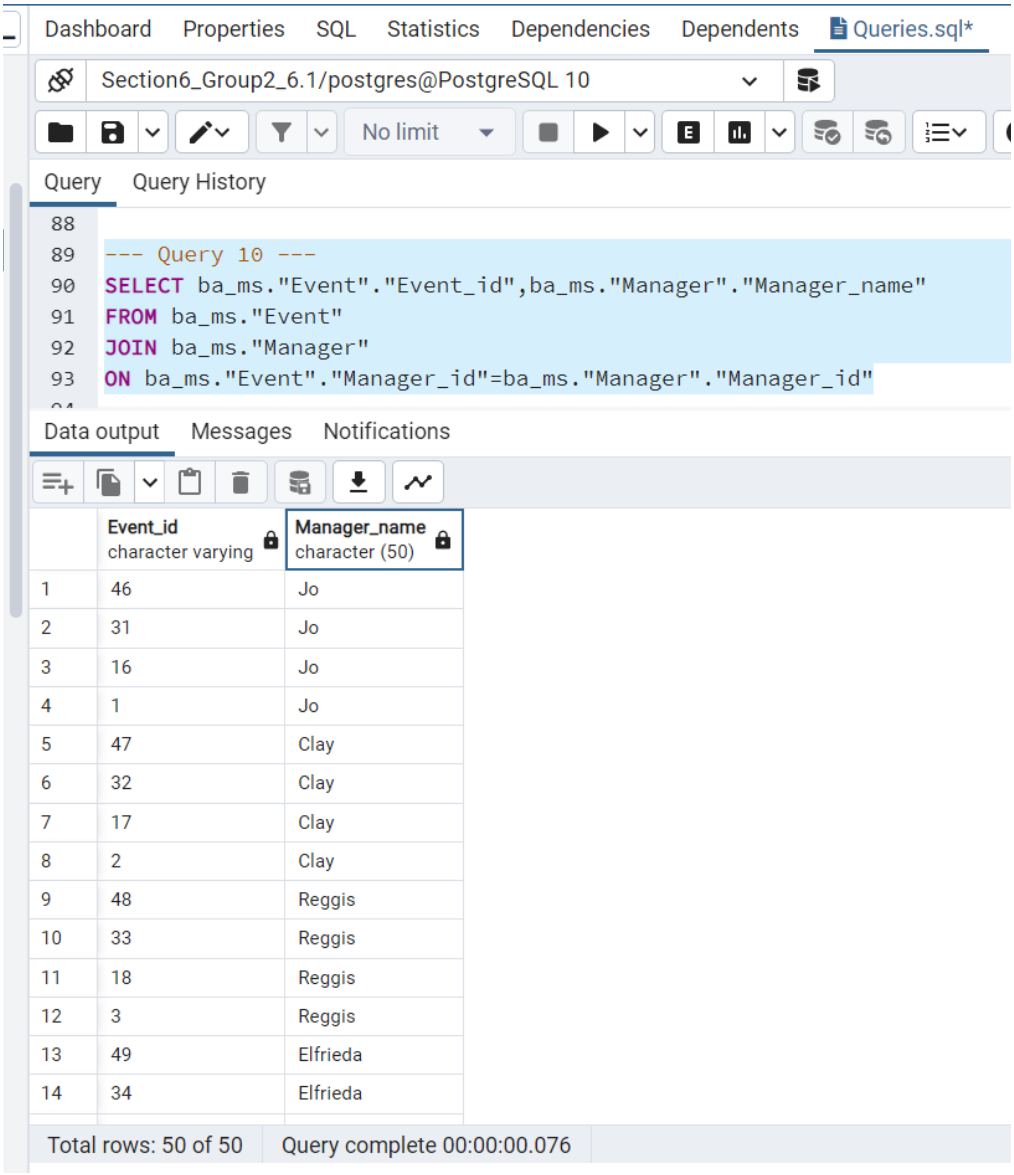
```
--- Query 9 ----
SELECT ba_ms."Trainers"."Trainer_id", ba_ms."Trainers"."Trainer_name",
ba_ms."Activity"."Activity_id"
FROM ba_ms."Activity"
JOIN ba_ms."Trainers"
ON ba_ms."Activity"."Trainer_id" = ba_ms."Trainers"."Trainer_id"
```



## Q10: Print manager name associated with event id.

## Ans:

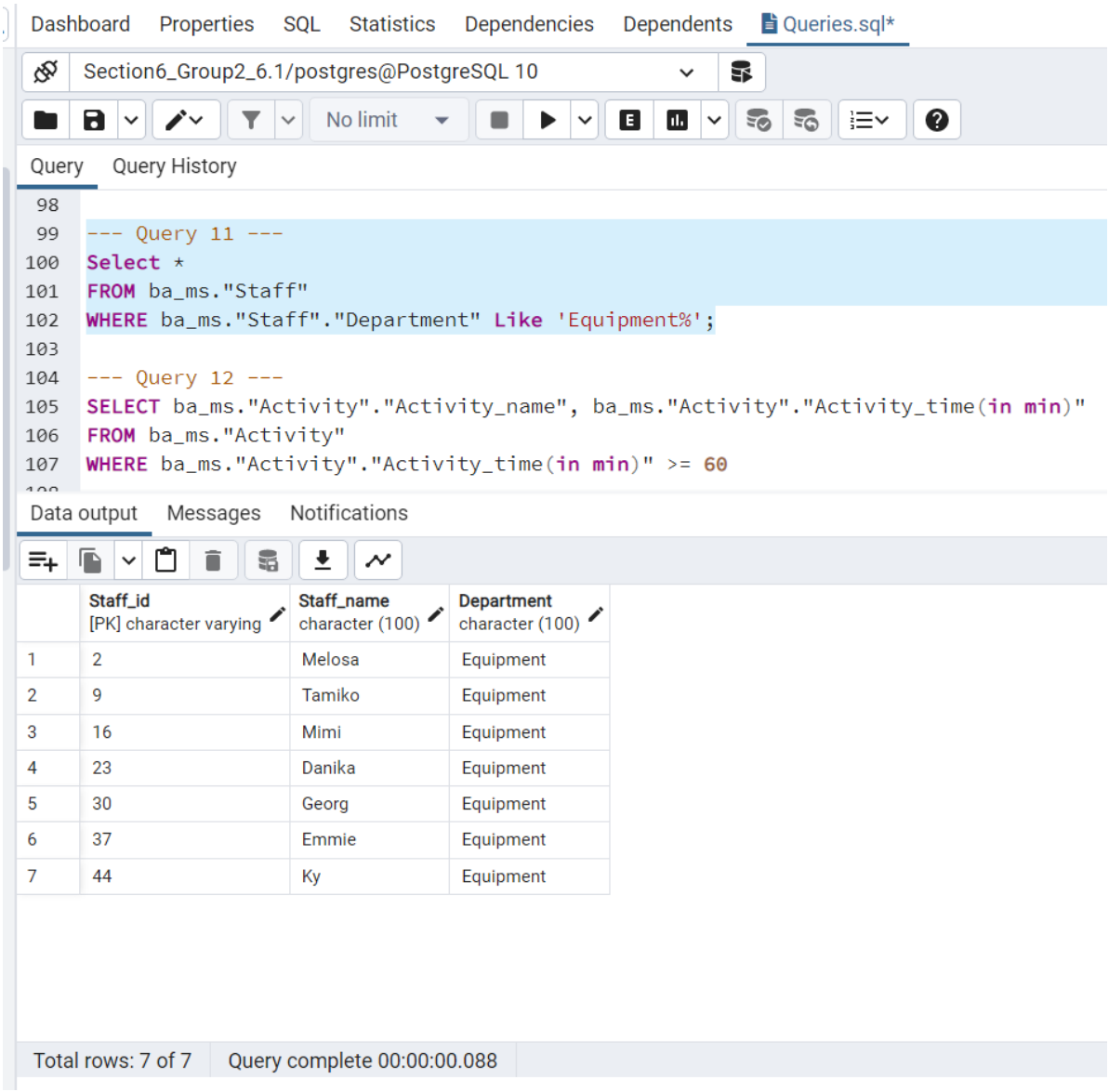
```
--- Query 10 ---
SELECT ba_ms."Event"."Event_id",ba_ms."Manager"."Manager_name"
FROM ba_ms."Event"
JOIN ba_ms."Manager"
ON ba_ms."Event"."Manager_id"=ba_ms."Manager"."Manager_id"
```



## Q11: Print names of staff members from equipment.

## Ans:

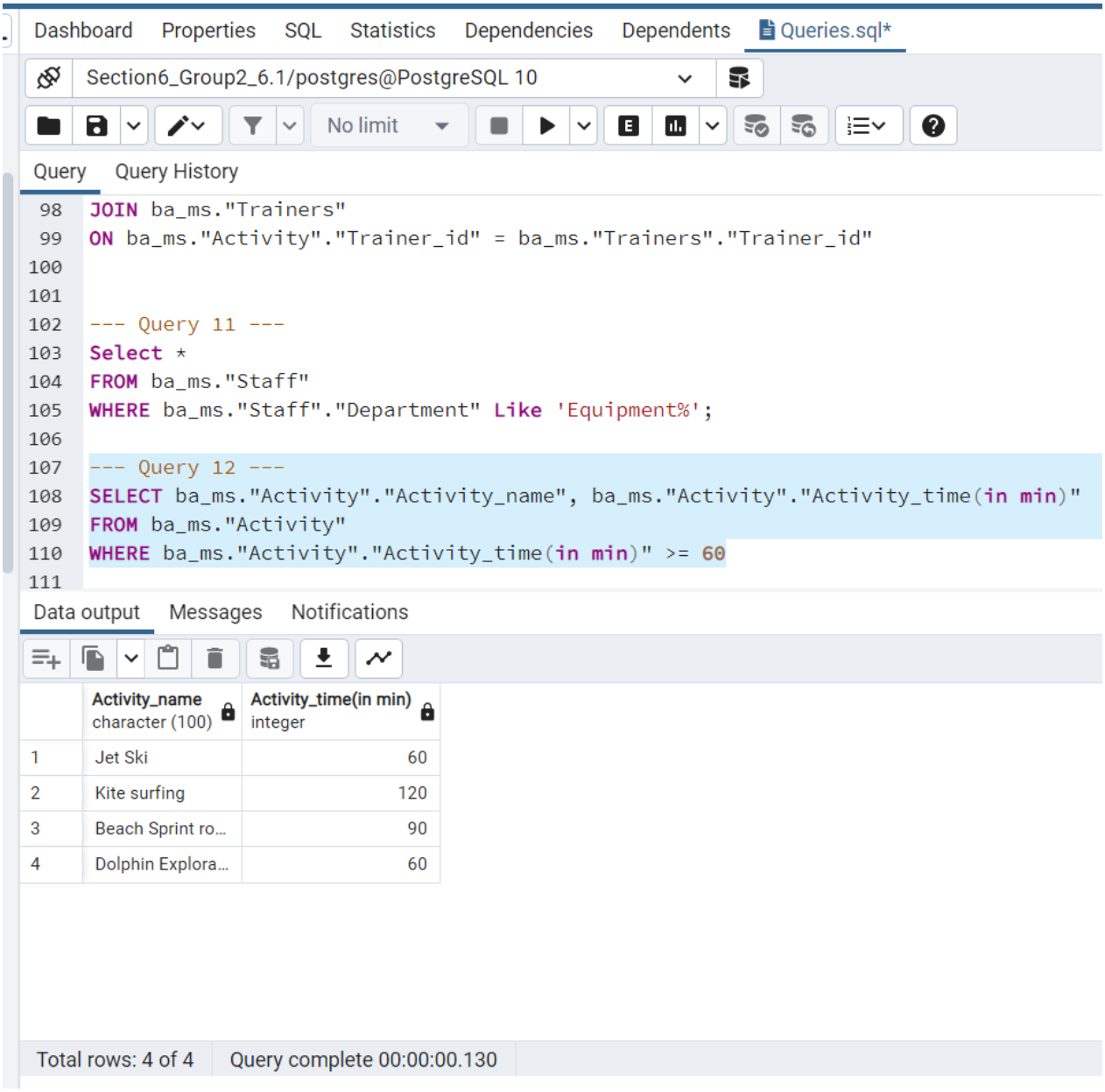
--- Query 11 ---
Select *
FROM ba_ms."Staff"
WHERE ba_ms."Staff"."Department" Like 'Equipment%';



## Q12: Print activity name whose activity time is greater or equal to 60 min.

## Ans:

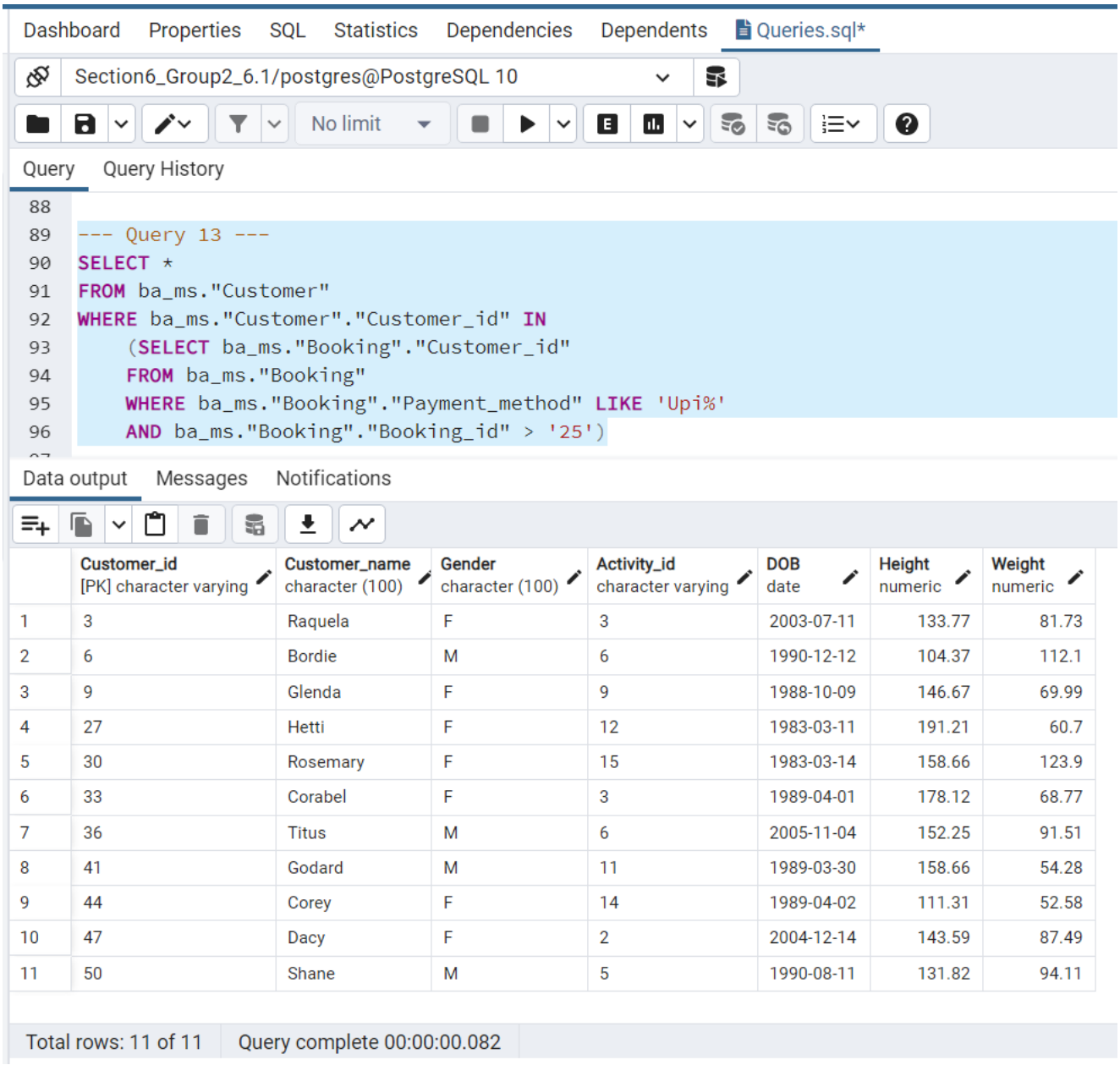
```
--- Query 12 ---
SELECT ba_ms."Activity"."Activity_name", ba_ms."Activity"."Activity_time(in min)"
FROM ba_ms."Activity"
WHERE ba_ms."Activity"."Activity_time(in min)" >= 60
```



**Q13: Display the details of customers with booking IDs greater than 25 who have done payments by UPI.**

**Ans:**

```
--- Query 13 ---
SELECT *
FROM ba_ms."Customer"
WHERE ba_ms."Customer"."Customer_id" IN
        (SELECT ba_ms."Booking"."Customer_id"
        FROM ba_ms."Booking"
        WHERE ba_ms."Booking"."Payment_method" LIKE 'Upi%'
        AND ba_ms."Booking"."Booking_id" > '25')
```

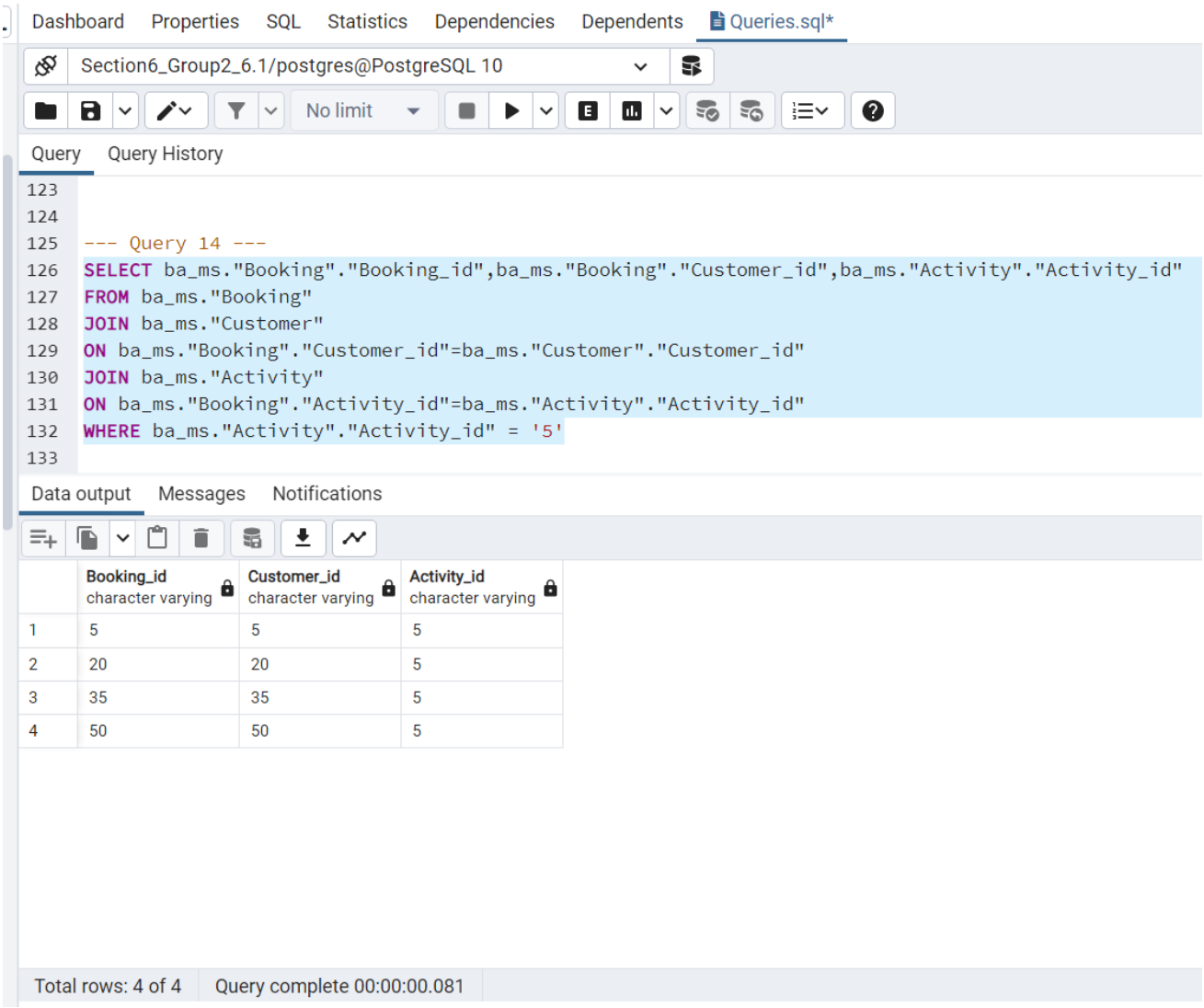



| | Customer_id [PK] character varying | Customer_name character (100) | Gender character (100) | Activity_id character varying | DOB date | Height numeric | Weight numeric |
|---|---|---|---|---|---|---|---|
| 1 | 3 | Raquela | F | 3 | 2003-07-11 | 133.77 | 81.73 |
| 2 | 6 | Bordie | M | 6 | 1990-12-12 | 104.37 | 112.1 |
| 3 | 9 | Glenda | F | 9 | 1988-10-09 | 146.67 | 69.99 |
| 4 | 27 | Hetti | F | 12 | 1983-03-11 | 191.21 | 60.7 |
| 5 | 30 | Rosemary | F | 15 | 1983-03-14 | 158.66 | 123.9 |
| 6 | 33 | Corabel | F | 3 | 1989-04-01 | 178.12 | 68.77 |
| 7 | 36 | Titus | M | 6 | 2005-11-04 | 152.25 | 91.51 |
| 8 | 41 | Godard | M | 11 | 1989-03-30 | 158.66 | 54.28 |
| 9 | 44 | Corey | F | 14 | 1989-04-02 | 111.31 | 52.58 |
| 10 | 47 | Dacy | F | 2 | 2004-12-14 | 143.59 | 87.49 |
| 11 | 50 | Shane | M | 5 | 1990-08-11 | 131.82 | 94.11 |

## Q14: Display booking ID and customer ID for an activity ID = 5

## Ans:

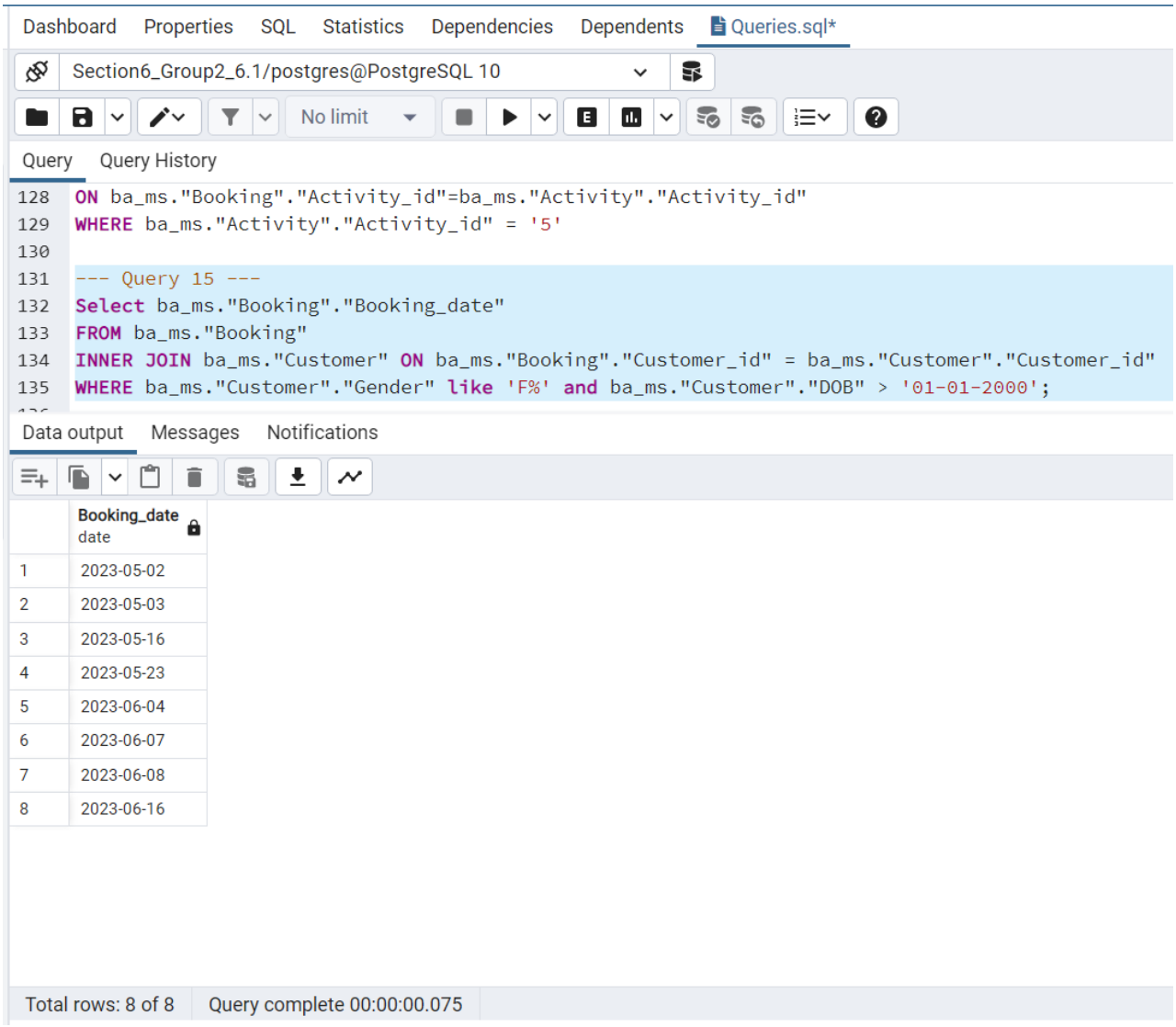
```
SELECT
ba_ms."Booking"."Booking_id",ba_ms."Booking"."Customer_id",ba_ms."Activity"."Activity_id"
FROM ba_ms."Booking"
JOIN ba_ms."Customer"
ON ba_ms."Booking"."Customer_id"=ba_ms."Customer"."Customer_id"
JOIN ba_ms."Activity"
ON ba_ms."Booking"."Activity_id"=ba_ms."Activity"."Activity_id"
WHERE ba_ms."Activity"."Activity_id" = '5'
```



## Q15: Find booking date of female customer whose dob is greater than 01 - 01 - 2000 ?

## Ans:

```
--- Query 15 ---
Select ba_ms."Booking"."Booking_date"
FROM ba_ms."Booking"
INNER JOIN ba_ms."Customer" ON ba_ms."Booking"."Customer_id" =
ba_ms."Customer"."Customer_id"
WHERE ba_ms."Customer"."Gender" like 'F%' and ba_ms."Customer"."DOB" > '01-01-2000';
```

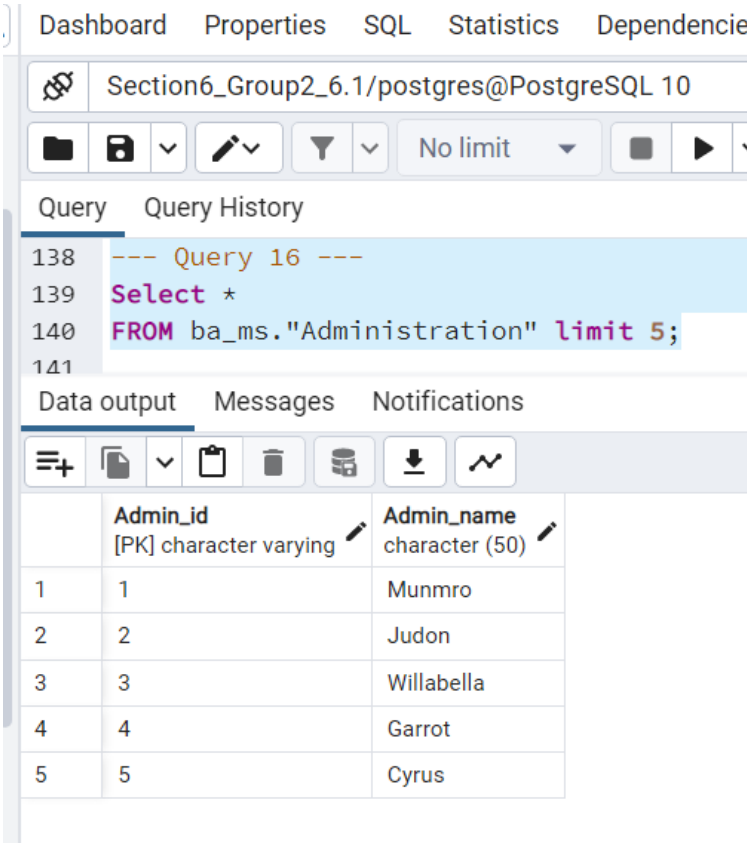


## Q16: Print the top 5 admin names.

## Ans:

--- Query 16 ---

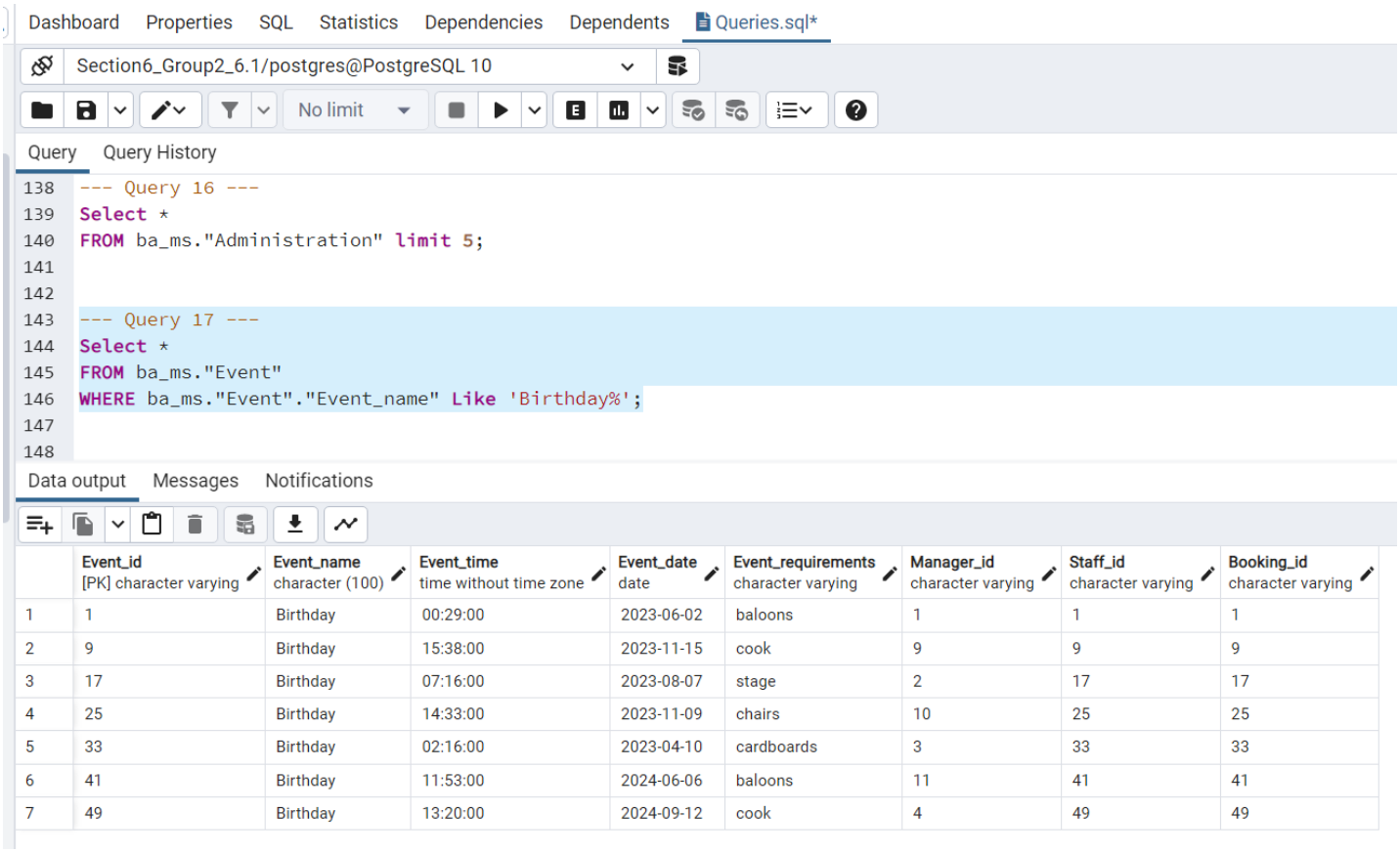
Select *
FROM ba_ms."Administration" limit 5;



## Q17: Print details of all the birthday events.

## Ans:

```
--- Query 17 ---
Select *
FROM ba_ms."Event"
WHERE ba_ms."Event"."Event_name" Like 'Birthday%';
```

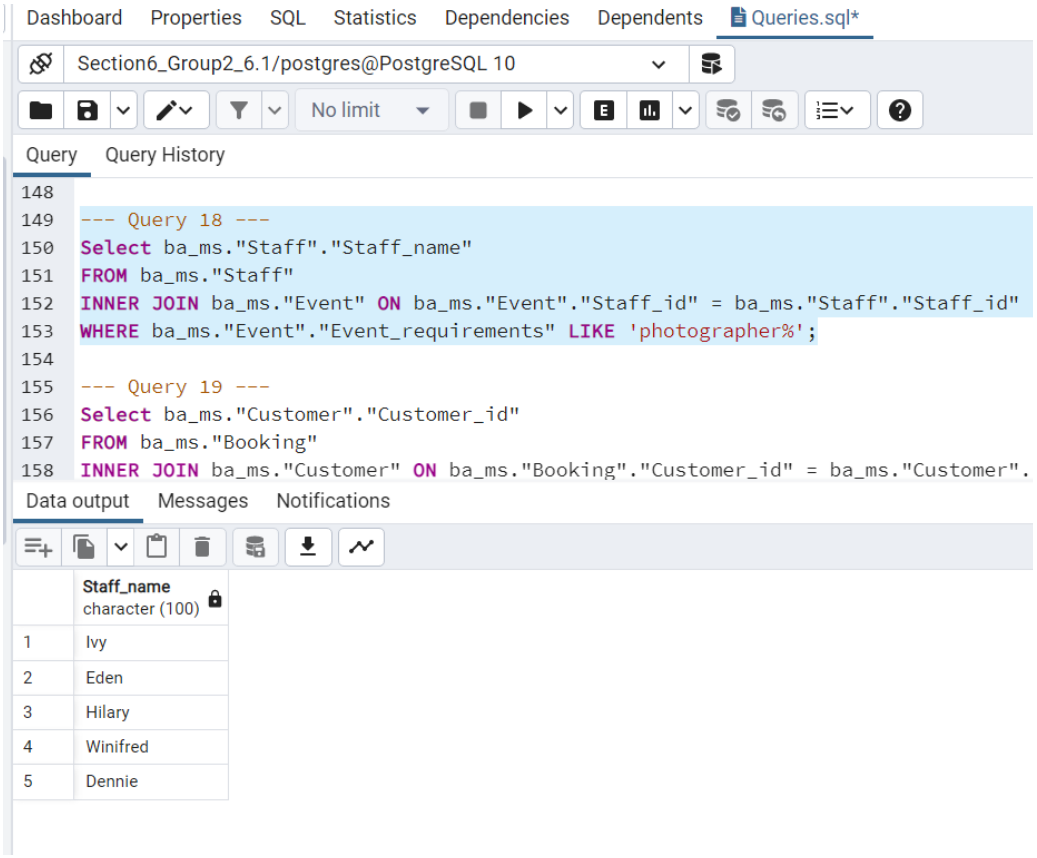



| | Event_id [PK] character varying | Event_name character (100) | Event_time time without time zone | Event_date date | Event_requirements character varying | Manager_id character varying | Staff_id character varying | Booking_id character varying |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | Birthday | 00:29:00 | 2023-06-02 | baloons | 1 | 1 | 1 |
| 2 | 9 | Birthday | 15:38:00 | 2023-11-15 | cook | 9 | 9 | 9 |
| 3 | 17 | Birthday | 07:16:00 | 2023-08-07 | stage | 2 | 17 | 17 |
| 4 | 25 | Birthday | 14:33:00 | 2023-11-09 | chairs | 10 | 25 | 25 |
| 5 | 33 | Birthday | 02:16:00 | 2023-04-10 | cardboards | 3 | 33 | 33 |
| 6 | 41 | Birthday | 11:53:00 | 2024-06-06 | baloons | 11 | 41 | 41 |
| 7 | 49 | Birthday | 13:20:00 | 2024-09-12 | cook | 4 | 49 | 49 |

## Q18: Find staff_name whose event requirement is a photographer.

## Ans:

```
--- Query 18 ---
Select ba_ms."Staff"."Staff_name"
FROM ba_ms."Staff"
INNER JOIN ba_ms."Event" ON ba_ms."Event"."Staff_id" = ba_ms."Staff"."Staff_id"
WHERE ba_ms."Event"."Event_requirements" LIKE 'photographer%';
```

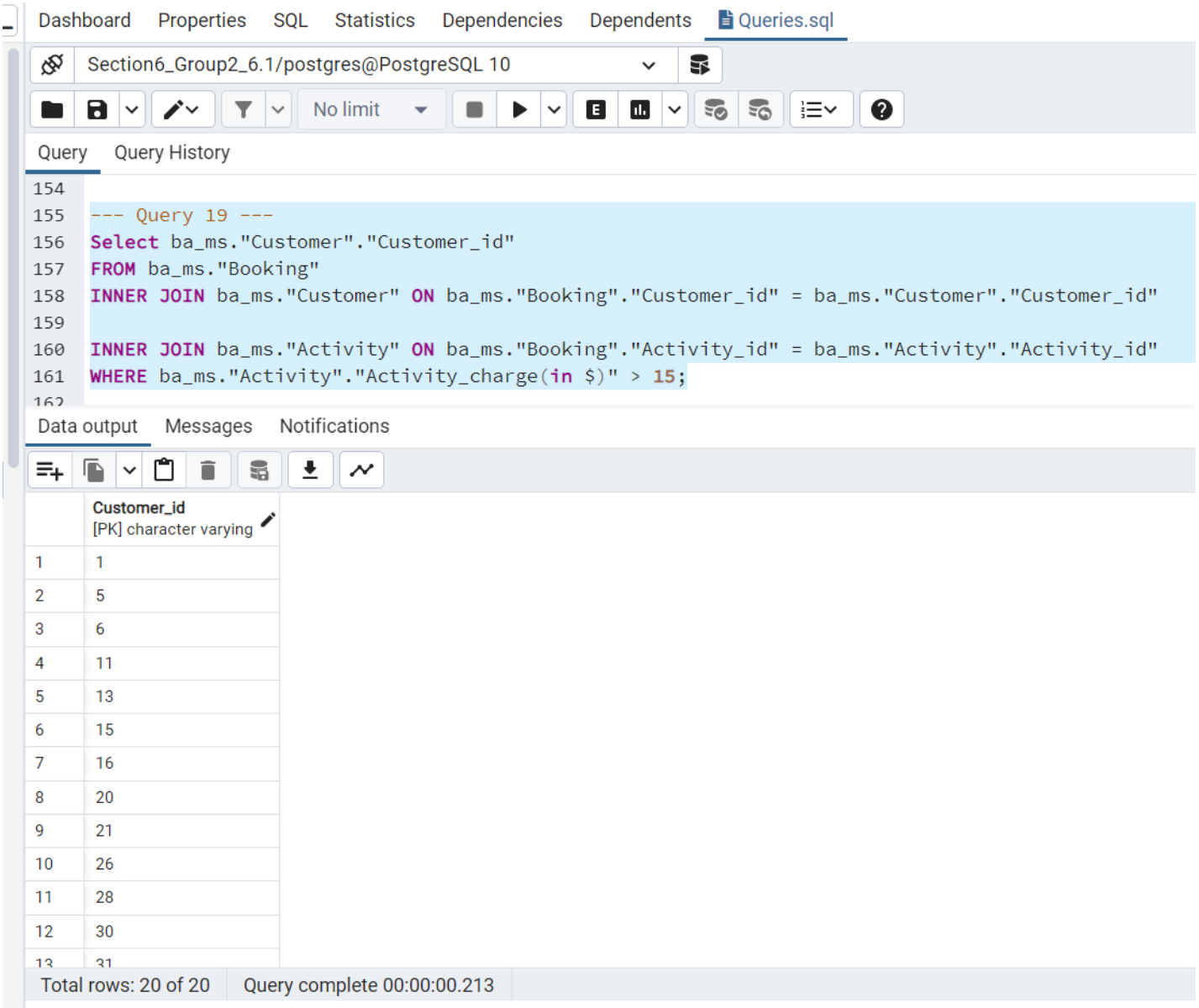


## Q19: Find the customer_id of booking done more than $15.

## Ans:

```
-- Query 19 --
Select ba_ms."Customer"."Customer_id"
FROM ba_ms."Booking"
INNER JOIN ba_ms."Customer" ON ba_ms."Booking"."Customer_id" =
ba_ms."Customer"."Customer_id"

INNER JOIN ba_ms."Activity" ON ba_ms."Booking"."Activity_id" =
ba_ms."Activity"."Activity_id"
WHERE ba_ms."Activity"."Activity_charge(in $)" > 15;
```



## Q20: Print manager_id, staff_id where the department is activity.

## Ans:

```
--- Query 20 ---
SELECT ba_ms."Manager"."Manager_id", ba_ms."Staff"."Staff_id"
FROM ba_ms."Manager"
JOIN ba_ms."Staff"
ON ba_ms."Manager"."Department"=ba_ms."Staff"."Department"
WHERE ba_ms."Manager"."Department" LIKE 'Activity%'
```