Overview

- **SELECT Statement**: Used to retrieve data using clauses like `WHERE`, `ORDER BY`, and `LIMIT`.
- **INSERT Statement**: Used to add new records into a table using single quotes for text values.
- **UPDATE Statement**: Used to modify existing records, requiring a `WHERE` clause to avoid changing every row.
- **DELETE Statement**: Used to remove records, requiring a `WHERE` clause to prevent deleting an entire table.
- **CRUD Acronym**: Represents Create (`INSERT`), Read (`SELECT`), Update (`UPDATE`), and Delete (`DELETE`).
- **Practical Exercises**: Structured tasks covering basic queries, modifications, and challenge problems for students.
- **Safety Practice**: Always run a `SELECT` query with the same `WHERE` condition before executing an `UPDATE` or `DELETE` statement.

---

1. SELECT Statement

The `SELECT` statement is used to retrieve data from a table.

Basic Syntax

```
SELECT column1, column2
FROM table_name;
```

Example 1: Display all customers

```
SELECT *
FROM customer;
```

`*` means all columns.

Example 2: Display specific columns

```
SELECT first_name, last_name, email
FROM customer;
```

Example 3: Display movies with their rental rates

```
SELECT title, rental_rate
FROM film;
```

Using `WHERE`

`WHERE` is used to retrieve records that meet a condition.

```sql
SELECT first_name, last_name
FROM customer
WHERE active = 1;
```

Using `ORDER BY`

```sql
SELECT title, rental_rate
FROM film
ORDER BY rental_rate DESC;
```

- `DESC` = highest to lowest
- `ASC` = lowest to highest

Using `LIMIT`

```sql
SELECT *
FROM film
LIMIT 10;
```

This displays only the first 10 records.

---

2. INSERT Statement

The `INSERT` statement is used to add a new record to a table.

Basic Syntax

```sql
INSERT INTO table_name (column1, column2, column3)
VALUES (value1, value2, value3);
```

Example

Suppose we want to add a new customer:

```sql
INSERT INTO customer
(first_name, last_name, email, address_id, store_id, active)
VALUES
('Ali', 'Khan', 'ali@example.com', 1, 1, 1);
```

Check the inserted record

```sql
SELECT *
FROM customer
```

```
WHERE email = 'ali@example.com';
```

> Important: When inserting text values, use single quotes like `'Ali'` and not `Ali`.

---

3. UPDATE Statement

The `UPDATE` statement is used to modify existing records.

Basic Syntax

```
UPDATE table_name
SET column = new_value
WHERE condition;
```

Example 1

Update a customer's email:

```
UPDATE customer
SET email = 'ali.khan@example.com'
WHERE customer_id = 600;
```

Example 2

Update a movie's rental rate:

```
UPDATE film
SET rental_rate = 3.99
WHERE film_id = 1;
```

Updating multiple columns

```
UPDATE customer
SET first_name = 'Muhammad',
    last_name = 'Ali'
WHERE customer_id = 600;
```

> ⚠️ Important: Always use `WHERE`.
>
> ```
> UPDATE customer
> SET active = 0;
> ```
>
> This will update every customer. Usually, you should specify the record:

```sql
UPDATE customer
SET active = 0
WHERE customer_id = 600;
```

---

4. DELETE Statement

The `DELETE` statement is used to remove records from a table.

Basic Syntax

```sql
DELETE FROM table_name
WHERE condition;
```

Example

```sql
DELETE FROM customer
WHERE customer_id = 600;
```

This removes the customer whose `customer_id` is `600`.

> ⚠️ Important: Always use `WHERE`.
>
> This is dangerous:
>
> ```sql
> DELETE FROM customer;
> ```
>
> It deletes all records from the `customer` table.
>
> Use:
>
> ```sql
> DELETE FROM customer
> WHERE customer_id = 600;
> ```

---

5. SELECT vs INSERT vs UPDATE vs DELETE

Statement	Purpose	Example
`SELECT`	Read or retrieve data	`SELECT * FROM film;`
`INSERT`	Add new data	`INSERT INTO film ...`
`UPDATE`	Modify existing data	`UPDATE film SET ...`
`DELETE`	Remove data	`DELETE FROM film WHERE ...`

Easy way to remember

CRUD:

- C — Create → `INSERT`
- R — Read → `SELECT`
- U — Update → `UPDATE`
- D — Delete → `DELETE`

---

6. Practical Exercises for Students

Exercise 1 — SELECT

Exercise 2 — INSERT

Exercise 3 — UPDATE

Exercise 4 — DELETE

Challenge

Write queries to:

1. Show all active customers.
2. Insert a new film into the `film` table.
3. Update a rental price.
4. Delete a customer only if a specific condition is met.

---

6. Practical Exercises for Students

Exercise 1 — SELECT

Display all films.

```
SELECT *
FROM film;
```

Exercise 2 — SELECT with WHERE

Display films with a rental rate greater than 3.

```
SELECT title, rental_rate
FROM film
WHERE rental_rate > 3;
```

Exercise 3 — INSERT

Insert a new customer with your own sample information.

Exercise 4 — UPDATE

Change the email address of the customer you inserted.

Exercise 5 — DELETE

Delete the customer you inserted.

Exercise 6 — Verification

After each operation, use `SELECT` to verify the result:

```
SELECT *
FROM customer
WHERE email = 'your_email@example.com';
```

> Important Safety Rule: Before executing `UPDATE` or `DELETE`, first run a `SELECT` with the same `WHERE` condition.
>
> ```
> SELECT *
> FROM customer
> WHERE customer_id = 600;
> ```
>
> If the correct record appears, then perform:
>
> ```
> UPDATE customer
> SET active = 0
> WHERE customer_id = 600;
> ```

Practice Exercises — PostgreSQL DVD Rental Database

Part A — SELECT Practice

Exercise 1: Display all films

Write a query to display all records from the `film` table.

```
SELECT *
FROM film;
```

Exercise 2: Select specific columns

Display the following information from the `film` table:

- Film title
- Release year
- Rental rate

Exercise 3: Find expensive films

Display films where the rental rate is greater than 3.

Exercise 4: Find short films

Display films whose length is less than 90 minutes.

Exercise 5: Find active customers

Display the first name, last name, and email of all active customers.

Exercise 6: Sort films

Display all films sorted by rental rate from highest to lowest.

Exercise 7: Display top 10 films

Display the first 10 films from the `film` table.

Exercise 8: Search by name

Find customers whose first name is `Mary`.

Part B — INSERT Practice

Exercise 9: Add a customer

Insert a new customer with:

- First name: Ali
- Last name: Khan
- Email: ali.khan@example.com
- Address ID: 1
- Store ID: 1
- Active: 1

Task: After inserting the record, use `SELECT` to verify it.

Exercise 10: Add another customer

Insert another customer using your own sample information.

Tasks:

1. Insert the customer.

2. Display the inserted customer.
3. Confirm that the information is correct.

Part C — UPDATE Practice

Exercise 11: Update email

Update the email address of the customer you inserted in Exercise 9.

Change it to:

`ali.new@example.com`

Then verify the change using `SELECT`.

Exercise 12: Update customer name

Change the last name of your inserted customer.

Example: `Khan` → `Ahmed`

Exercise 13: Update rental rate

Find a film with `film_id = 1`.

Change its rental rate to `3.99` and verify the result.

Exercise 14: Update multiple columns

For your inserted customer, update:

- First name
- Last name
- Email

using a single `UPDATE` statement.

Part D — DELETE Practice

Exercise 15: Delete a customer

Delete the customer that you inserted in Exercise 9.

Important: Use the appropriate `WHERE` condition.

Then verify that the customer has been deleted.

Exercise 16: Delete another test record

Insert a new test customer and then delete that customer.

Tasks:

1. Insert the customer.
2. Find the customer using `SELECT`.
3. Delete the customer.
4. Use `SELECT` again to verify that the record no longer exists.

Part E — Important WHERE Practice

For each task, first write a `SELECT` query to identify the records.

Exercise 17

Find all films with a rental rate greater than 4.

Exercise 18

Find all films with a rental rate equal to 2.99.

Exercise 19

Find customers whose first name starts with `A`.

Exercise 20

Find customers whose last name is `Smith`.

Exercise 21

Find films with a length greater than 120 minutes.

Part F — CRUD Practical Task

Complete the following task independently.

Task 1 — Customer CRUD

Perform all four CRUD operations on the `customer` table.

- C — Create: Insert a new customer.
- R — Read: Display the customer's information.
- U — Update: Change the customer's email.
- D — Delete: Delete the customer.

Expected sequence

```
INSERT
   ↓
```

```
SELECT
   ↓
UPDATE
   ↓
SELECT
   ↓
DELETE
   ↓
SELECT
```

Part G — Film Management Task

Perform the following operations on the `film` table.

1. Display all films.
2. Display only `title`, `release_year`, and `rental_rate`.
3. Find films with rental rate greater than 3.
4. Find films longer than 120 minutes.
5. Sort films by rental rate.
6. Update the rental rate of one test film.
7. Verify the update.

Part H — Challenge Tasks

Try these without looking at previous examples.

Challenge 1

Find all customers whose first name starts with `J`.

Challenge 2

Find all films with:

- Rental rate greater than 2
- Length greater than 100

Challenge 3

Display the 10 films with the highest rental rate.

Challenge 4

Find customers whose email contains `gmail`.

Challenge 5

Find films released after the year 2005.

Challenge 6

Update the rental rate of a selected film and verify the change.

Challenge 7

Create a test customer, update the customer's information, and finally delete the customer.

Part I — Safety Practice

Before every `UPDATE` or `DELETE`, students should first run a `SELECT`.

Step 1 — Check the record

```
SELECT *
FROM customer
WHERE customer_id = 600;
```

Step 2 — Update

```
UPDATE customer
SET active = 0
WHERE customer_id = 600;
```

Step 3 — Verify

```
SELECT *
FROM customer
WHERE customer_id = 600;
```

> Remember: First `SELECT` → Check → `UPDATE`/`DELETE` → `SELECT` again.
>
> This habit helps prevent accidental changes to multiple records.

Summary

CRUD operations are the foundation of database interaction in PostgreSQL. By learning `SELECT`, `INSERT`, `UPDATE`, and `DELETE`, we can manage data efficiently and safely in real-world database systems.

Key reminder

- Use `SELECT` to read data.
- Use `INSERT` to create records.
- Use `UPDATE` to modify records.
- Use `DELETE` to remove records.

- Always use a `WHERE` clause when updating or deleting data.
