User story: Seat management for coworking system

Requirement: As a Product Owner I want users to be able to reserve coworking spaces for a specific session coworking spaces for a specific session to facilitate the management of coworking spaces and space occupancy and improve the user experience.

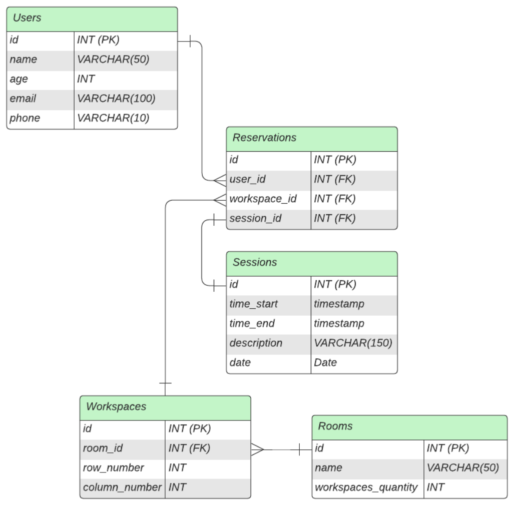
Task: As a backend developer I want to create the SQL data model to support this requirement and implement it in requirement and implement it in PostgreSQL.

Deliverables: Sent to the milton.loaiza@riwi.io

● Public link with confluence documentation.

- *this document*

● Image or PDF with entity-relationship diagram.



● Public link to the repository with SQL scripts.

- SamuelSml8/COWORKING-DB

● Access data to the database implemented in the cloud.

- connection to PostgreSQL and Clever Cloud
  - **HOST:** bjy2odltfxx6fboyf9ny-postgresql.services.clever-cloud.com
  - **DATABASE NAME:** bjy2odltfxx6fboyf9ny
  - **USER:** uyvlvfzmq6v8vxlqrsuf
  - **PASSWORD:** *value in the email sent*
  - **PORT:** 50013

SCRIPTS

- SCRIPTS
  - Workspaces available in a room in a session x

Workspaces available in a room in a session x

```sql
SELECT w.id, w.row_number, w.column_number
FROM workspaces w
LEFT JOIN reservations r ON w.id = r.workspace_id
WHERE w.room_id = :room_id
  AND (r.session_id <> :session_id OR r.session_id IS NULL);
```

Explanation:

- **SELECT w.id, w.row_number, w.column_number:** Selects the columns `id`, `row_number`, and `column_number` from the `workspaces` table.
- **FROM workspaces w:** Indicates that the main table is `workspaces` and assigns it the alias `w`.
- **LEFT JOIN reservations r ON w.id = r.workspace_id:** Performs a left join between the `workspaces` table and the `reservations` table based on the condition that the workspace `id` in `workspaces` should match the `workspace_id` in `reservations`. This ensures that all workspaces are selected, even those without reservations.
- **WHERE w.room_id = :room_id:** Filters the workspaces that belong to a specific room, indicated by the parameter `:room_id`.
- **AND (r.session_id <>**

  OR r.session_id IS NULL): Filters the workspaces that are not reserved for the specific session (`:session_id`) or that have no reservation (when `r.session_id` is `NULL`).

Workspaces occupied of a room in x session

```sql
SELECT w.id, w.row_number, w.column_number
FROM workspaces w
JOIN reservations r ON w.id = r.workspace_id
WHERE w.room_id = :room_id
  AND r.session_id = :session_id;
```

Explanation:

- **SELECT w.id, w.row_number, w.column_number:** Selects the columns `id`, `row_number`, and `column_number` from the `workspaces` table.
- **FROM workspaces w:** Indicates that the main table is `workspaces` and assigns it the alias `w`.
- **JOIN reservations r ON w.id = r.workspace_id:** Performs an inner join between the `workspaces` table and the `reservations` table based on the condition that the workspace `id` in `workspaces` should match the `workspace_id` in `reservations`. This selects only the workspaces that have a reservation.
- **WHERE w.room_id = :room_id:** Filters the workspaces that belong to a specific room (`:room_id`).
- **AND r.session_id = :session_id:** Filters the workspaces that are reserved for the specific session (`:session_id`).

Sessions in order by most occupied.

```sql
SELECT s.id, s.time_start, s.time_end, s.description, s.date, COUNT(r.id) AS spaces_occupied
FROM sessions s
    LEFT JOIN reservations r ON s.id = r.session_id
GROUP BY
    s.id,
    s.time_start,
    s.time_end,
    s.description,
    s.date
ORDER BY spaces_occupied DESC;
```

Explanation:

- **SELECT s.id , s.time_start, s.time_end, s.description, s.date, COUNT(r.id) AS spaces_occupied:** Selects the columns `id`, `time_start`, `time_end`, `description`, `date` from the `sessions` table and counts the number of reservations (`r.id`), aliasing it as `spaces_occupied`.
- **FROM sessions s:** Indicates that the main table is `sessions` and assigns it the alias `s`.
- **LEFT JOIN reservations r ON s.id = r.session_id:** Performs a left join between the `sessions` table and the `reservations` table based on the condition that the session `id` in `sessions` should match the `session_id` in `reservations`. This ensures that all sessions are selected, even those without reservations.
- **GROUP BY s.id , s.time_start, s.time_end, s.description, s.date:** Groups the results by all the selected columns from `sessions`.
- **ORDER BY spaces_occupied DESC:** Orders the sessions by the number of occupied workspaces (`spaces_occupied`) in descending order, showing the most occupied sessions first.

Sessions with order by most available.

```sql
SELECT s.id, s.time_start, s.time_end, s.description, s.date,
       (SELECT COUNT(w.id) FROM workspaces w WHERE w.room_id = :room_id) - COUNT(r.id) AS spaces_available
FROM sessions s
LEFT JOIN reservations r ON s.id = r.session_id
WHERE s.date = :session_date  -- Optional: Filter by specific date if needed
GROUP BY s.id, s.time_start, s.time_end, s.description, s.date
ORDER BY spaces_available DESC;
```

Explanation:

- **SELECT s.id , s.time_start, s.time_end, s.description, s.date,**: Selects the columns `id`, `time_start`, `time_end`, `description`, `date` from the `sessions` table.
- **(SELECT COUNT(w.id) FROM workspaces w WHERE w.room_id =**

) - COUNT(r.id) AS spaces_available: Calculates the number of available workspaces by subtracting the number of reservations (`COUNT(r.id)`) from the total number of workspaces in the room (`COUNT(w.id)`).
- **FROM sessions s:** Indicates that the main table is `sessions` and assigns it the alias `s`.
- **LEFT JOIN reservations r ON s.id = r.session_id:** Performs a left join between the `sessions` table and the `reservations` table based on the condition that the session `id` in `sessions` should match the `session_id` in `reservations`. This ensures that all sessions are selected, even those without reservations.
- **WHERE s.date = :session_date:** (Optional) Filters the sessions by a specific date (`:session_date`).
- **GROUP BY s.id , s.time_start, s.time_end, s.description, s.date:** Groups the results by all the selected columns from `sessions`.
- **ORDER BY spaces_available DESC:** Orders the sessions by the number of available workspaces (`spaces_available`) in descending order, showing the most available sessions first.

List of workspaces assigned to a user.

```sql
SELECT w.id, w.row_number, w.column_number
FROM workspaces w
JOIN reservations r ON w.id = r.workspace_id
WHERE r.user_id = :user_id;
```

Explanation:

- **SELECT w.id, w.row_number, w.column_number:** Selects the columns `id`, `row_number`, and `column_number` from the `workspaces` table.
- **FROM workspaces w:** Indicates that the main table is `workspaces` and assigns it the alias `w`.
- **JOIN reservations r ON w.id = r.workspace_id:** Performs an inner join between the `workspaces` table and the `reservations` table based on the condition that the workspace `id` in `workspaces` should match the `workspace_id` in `reservations`. This selects only the workspaces that have a reservation.
- **WHERE r.user_id = :user_id:** Filters the reservations where the `user_id` matches the parameter `:user_id`.

List of workspaces assigned to a session.

```sql
SELECT w.id, w.row_number, w.column_number
FROM workspaces w
JOIN reservations r ON w.id = r.workspace_id
WHERE r.session_id = :session_id;
```

Explanation:

- **SELECT w.id, w.row_number, w.column_number:** Selects the columns `id`, `row_number`, and `column_number` from the `workspaces` table.
- **FROM workspaces w:** Indicates that the main table is `workspaces` and assigns it the alias `w`.
- **JOIN reservations r ON w.id = r.workspace_id:** Performs an inner join between the `workspaces` table and the `reservations` table based on the condition that the workspace `id` in `workspaces` should match the `workspace_id` in `reservations`. This selects only the workspaces that have a reservation.
- **WHERE r.session_id = :session_id:** Filters the reservations where the `session_id` matches the parameter `:session_id`.

Social networks

GitHub: SamuelSml8

LinkedIn: Samuel Vera Miranda

Thank you for watching. 🫡💜