**A9 – Main Accesses to the database and transactions**

This artefact shows the main accesses to the database in the Project Management application. All the main accesses are described with the corresponding SQL code and web resource reference. It also includes all the required transactions for this application.

1. **Main Accesses**
   1. M01 Registration and Authentication

|  |  |
| --- | --- |
| SQL101 | Creates new user in the platform |
| Web resource | R105 |
| INSERT INTO User (e\_mail, password, URL, username)  VALUES ($e\_mail, $password, $URL, $username) | |

|  |  |
| --- | --- |
| SQL102 | Verifies existence of e-mail in the user table |
| Web resource | R102 |
| CREATE FUNCTION check\_exist\_email() RETURNS TRIGGER AS  $BODY$  BEGIN  IF EXISTS (SELECT e\_mail FROM "User" WHERE NEW.e\_mail = e\_mail) THEN  RAISE EXCEPTION 'The e-mail you are trying to use is already registred'  END IF;  RETURN NEW;  END  $BODY$  LANGUAGE plpgsql;    CREATE TRIGGER check\_exist\_email  BEFORE INSERT ON "User"  FOR EACH ROW  EXECUTE PROCEDURE check\_exist\_user(); | |

|  |  |
| --- | --- |
| SQL103 | Checks what is the password associated to a certain e-mail in the user table |
| Web resource | R102 |
| SELECT password FROM "User"  WHERE user.e\_mail = $userEmail | |

* 1. M02 User Pages

|  |  |
| --- | --- |
| SQL201 | All user’s projects |
| Web resource | R113 |
| SELECT \*  FROM Project  INNER JOIN Project\_team ON Project.id=Project\_team.id\_project  WHERE Project.id\_coordinator=$user OR Project\_team.id\_user=$user | |

|  |  |
| --- | --- |
| SQL202 | All user’s personal events |
| Web resource | R107 |
| SELECT \* FROM Personal\_event  WHERE Personal\_event.id\_user=$user | |

|  |  |
| --- | --- |
| SQL203 | All user’s board meetings |
| Web resource | R107 |
| SELECT \*  FROM Meeting  INNER JOIN Board\_team ON Meeting.id\_board=Board\_team.id\_board  WHERE Board\_team.id\_worker=$user | |

|  |  |
| --- | --- |
| SQL204 | User’s personal info |
| Web resource | R106 |
| SELECT \* FROM "User"  WHERE User.id=$user | |

|  |  |
| --- | --- |
| SQL205 | Search projects |
| Web resource | R115 |
| SELECT id, name, description, start\_date, ts\_rank\_cd(textsearch, query) AS rank  FROM Project, to\_tsquery($search) AS query, to\_tsvector(name || ‘ ‘ || description) AS textsearch  WHERE query @@ textsearch\\ORDER BY rank DESC; | |

|  |  |
| --- | --- |
| SQL206 | Add personal event |
| Web resource | \*Resource not specified |
| INSERT INTO Personal\_event (date, place, name, id\_user)  VALUES ($date, $place, $name, $id\_user) | |

|  |  |
| --- | --- |
| SQL207 | Add new project |
| Web resource | R202 |
| INSERT INTO Project (description, name, id\_coordinator, privacy)  VALUES ($description, $name, $id\_coordinator, $privacy) | |

|  |  |
| --- | --- |
| SQL208 | Change user’s full name |
| Web resource | R109 |
| UPDATE User  SET full\_name=$full\_name | |

|  |  |
| --- | --- |
| SQL209 | Change user’s password |
| Web resource | R109 |
| UPDATE User  SET password=$password | |

|  |  |
| --- | --- |
| SQL210 | Change profile picture |
| Web resource | R109 |
| INSERT INTO Profile\_picture (id\_user, path)  VALUES ($id\_user, $path) | |

|  |  |
| --- | --- |
| SQL211 | Mark notification as read/unread |
| Web resource | \*Resource not specified |
| UPDATE Notification  SET read=$read | |

* 1. M03 Project Pages



|  |  |
| --- | --- |
| SQL301 | Project boards |
| Web resource | R213 |
| SELECT \*  FROM Board  WHERE Board.id\_project=$projectId | |

|  |  |
| --- | --- |
| SQL302 | Project information |
| Web resource | R203 |
| SELECT \* FROM Project  WHERE Project.id=$projectId | |

|  |  |
| --- | --- |
| SQL303 | Project workers and information |
| Web resource | R205 |
| SELECT User.full\_name, Profile\_picture.path, User.e\_mail  FROM "User"  INNER JOIN Profile\_picture ON Profile\_picture.id\_user=$user  WHERE Project.id=$projectId AND Profile\_picture.path=SELECT(MAX(Profile\_picture.id)) | |

|  |  |
| --- | --- |
| SQL304 | Project events |
| Web resource | R204 |
| SELECT \*  FROM Meeting  INNER JOIN Board ON Board.id=Meeting.id\_board  WHERE Board.id\_project=$projectId | |

|  |  |
| --- | --- |
| SQL305 | Project forum messages |
| Web resource | R212 |
| SELECT \*  FROM Message  WHERE Message.id\_project=$projectId AND Message.date >= DATEADD(day, -7, GETDATE()) | |

|  |  |
| --- | --- |
| SQL306 | Search users |
| Web resource | R206 |
| SELECT username, full\_name, e\_mail FROM User  INNER JOIN Project\_team ON User.id=Project\_team.id\_user  WHERE Project\_team.id\_project=$id\_project AND (username LIKE %’$search%’ OR full\_name LIKE %’$search%’);  ORDER BY username; | |

|  |  |
| --- | --- |
| SQL307 | Search boards |
| Web resource | R214 |
| SELECT id, name, description ts\_rank\_cd(textsearch, query) AS rank  FROM Board, to\_tsquery($search) AS query, to\_tsvector(name || ‘ ‘ || description) AS textsearch  WHERE query @@ textsearch\\ AND id\_project=$id\_project  ORDER BY rank DESC; | |

|  |  |
| --- | --- |
| SQL308 | Add new board |
| Web resource | R302 |
| INSERT INTO Board (description, name, id\_creator, id\_project)  VALUES ($description, $name, $id\_creator, $id\_project) | |

|  |  |
| --- | --- |
| SQL309 | Archive/close project |
| Web resource | \*Resource not specified |
| UPDATE Project  SET project\_state=$project\_state | |

|  |  |
| --- | --- |
| SQL310 | Add new worker |
| Web resource | R209 |
| INSERT INTO Project\_team (id\_user, id\_project)  VALUES ($id\_user, $id\_project) | |

|  |  |
| --- | --- |
| SQL311 | Change project description |
| Web resource | \*Resource not specified |
| UPDATE Project  SET description=$description | |

|  |  |
| --- | --- |
| SQL312 | Add project end date |
| Web resource | \*Resource not specified |
| UPDATE Project  SET end\_date=$end\_date | |

|  |  |
| --- | --- |
| SQL313 | Change project picture |
| Web resource | \*Resource not specified |
| INSERT INTO Project\_picture (id\_project, path)  VALUES ($id\_project, $path) | |

|  |  |
| --- | --- |
| SQL314 | Send message to forum |
| Web resource | R211 |
| INSERT INTO Message (message, id\_user, id\_project)  VALUES ($message, $id\_user, $id\_project) | |

|  |  |
| --- | --- |
| SQL314 | Trigger – no 2 boards on the same project can have the same name |
| Web resource | R302 |
| CREATE FUNCTION check\_exist\_board() RETURNS TRIGGER AS  $BODY$  BEGIN  IF EXISTS (SELECT name FROM Board WHERE Board.id\_project=$projectId AND NEW.name=name) THEN  RAISE EXCEPTION 'This project already has a board with that name'  END IF;  RETURN NEW;  END  $BODY$  LANGUAGE plpgsql;    CREATE TRIGGER check\_exist\_board  BEFORE INSERT ON Board  FOR EACH ROW  EXECUTE PROCEDURE check\_exist\_board(); | |

|  |  |
| --- | --- |
| SQL315 | Trigger – a user can’t be added to a project twice |
| Web resource | R209 |
| CREATE FUNCTION check\_exist\_worker() RETURNS TRIGGER AS  $BODY$  BEGIN  IF EXISTS (SELECT name FROM Project\_team WHERE Project\_team.id\_project=$projectId AND NEW.id\_worker=id\_worker) THEN  RAISE EXCEPTION 'This user already belongs to this team'  END IF;  RETURN NEW;  END  $BODY$  LANGUAGE plpgsql;    CREATE TRIGGER check\_exist\_worker  BEFORE INSERT ON Project\_team  FOR EACH ROW  EXECUTE PROCEDURE check\_exist\_worker(); | |

|  |  |
| --- | --- |
| SQL316 | Add user to contact list |
| Web resource | R111 |
| INSERT INTO Contact (id\_user, id\_contact)  VALUES ($id\_user, $id\_contact) | |

|  |  |
| --- | --- |
| SQL317 | Trigger – can’t add a user to contact list twice |
| Web resource | R111 |
| CREATE FUNCTION check\_exist\_contact() RETURNS TRIGGER AS  $BODY$  BEGIN  IF EXISTS (SELECT name FROM Contact WHERE Contact.id\_user=$userId AND NEW.id\_contact=id\_contact AND NEW.id\_contact=$userId) THEN  RAISE EXCEPTION 'This user already is on your contact list'  END IF;  RETURN NEW;  END  $BODY$  LANGUAGE plpgsql;    CREATE TRIGGER check\_exist\_contact  BEFORE INSERT ON Contact  FOR EACH ROW  EXECUTE PROCEDURE check\_exist\_contact(); | |

* 1. M04 Board pages

|  |  |
| --- | --- |
|  |  |
|  |  |
|  | |

|  |  |
| --- | --- |
| SQL401 | Board tasks |
| Web resource | R213 |
| SELECT \* FROM Task  WHERE Task.id\_board=$boardId | |

|  |  |
| --- | --- |
| SQL402 | Last 5 task updates |
| Web resource | \* Resource not specified |
| SELECT TOP 5 Comment.id, File.id, Progress\_update.id  FROM Task  INNER JOIN Comment ON Task.id=Comment.id\_task  INNER JOIN File ON Task.id=File.id\_task  INNER JOIN Task.id=Progress\_update.id\_task  WHERE Task.id=$taskId | |

|  |  |
| --- | --- |
| SQL403 | Search tasks |
| Web resource | R306 |
| SELECT id, name, description ts\_rank\_cd(textsearch, query) AS rank  FROM Task, to\_tsquery($search) AS query, to\_tsvector(name || ‘ ‘ || description) AS textsearch  WHERE query @@ textsearch\\ ORDER BY rank DESC; | |

|  |  |
| --- | --- |
| SQL404 | Add board meeting |
| Web resource | \* Resource not specified |
| INSERT INTO Meeting (date, name, place, id\_board)  VALUES ($date, $name, $place, $id\_board) | |

|  |  |
| --- | --- |
| SQL405 | Add new worker to board |
| Web resource | R304 |
| INSERT INTO Board\_team (id\_board, id\_user)  VALUES ($id\_board, $id\_user) | |

|  |  |
| --- | --- |
| SQL406 | Send notification |
| Web resource | \* Resource not specified |
| INSERT INTO Notification (id\_user, notification)  VALUES ($id\_user, $notification) | |

* 1. M05 Task page

|  |  |
| --- | --- |
|  |  |
|  |  |
|  | |

|  |  |
| --- | --- |
| SQL501 | Task information |
| Web resource | R405 |
| SELECT \*  FROM Task  WHERE Task.id=$taskId | |

|  |  |
| --- | --- |
| SQL502 | Insert new progress update |
| Web resource | R404 |
| INSERT INTO Progress\_update (new\_value, id\_user, id\_task)  VALUES ($new\_value, $id\_user, $id\_task) | |

|  |  |
| --- | --- |
| SQL503 | Trigger – update task progress |
| Web resource | R404 |
| CREATE FUNCTION update\_task\_progress() RETURNS TRIGGER AS  $BODY$  BEGIN  UPDATE Task  SET Task.progress=Progress\_update.new\_value  WHERE Task.id=Progress\_update.id\_task  END  $BODY$  LANGUAGE plpgsql;  CREATE TRIGGER update\_task\_progress  AFTER INSERT ON Progress\_update  FOR EACH ROW  Execute PROCEDURE update\_task\_progress(); | |

|  |  |
| --- | --- |
| SQL504 | Upload new file |
| Web resource | R404 |
| INSERT INTO File (path, description, id\_user, id\_task)  VALUES ($path, $description, $id\_user, $id\_task) | |

|  |  |
| --- | --- |
| SQL505 | Comment task |
| Web resource | R404 |
| INSERT INTO Comment (comment, id\_user, id\_task)  VALUES ($comment, $id\_user, $id\_task) | |

|  |  |
| --- | --- |
| SQL506 | Archive/close task |
| Web resource | R404 |
| UPDATE Task  SET task\_state=$task\_state | |







* 1. Homepage and administration

|  |  |
| --- | --- |
| SQL601 | All public projects |
| Web resource | \* Resource not specified |
| SELECT name FROM Project  WHERE Project.privacy=1 | |

|  |  |
| --- | --- |
| SQL602 | Search projects |
| Web resource | R701 |
| SELECT id, name, description, start\_date, ts\_rank\_cd(textsearch, query) AS rank  FROM Project, to\_tsquery($search) AS query, to\_tsvector(name || ‘ ‘ || description) AS textsearch  WHERE query @@ textsearch\\ ORDER BY rank DESC; | |

|  |  |
| --- | --- |
| SQL603 | Set user as administrator |
| Web resource | R507 |
| UPDATE User  SET administrator=$administrator | |

|  |  |
| --- | --- |
| SQL604 | Delete user |
| Web resource | R509 |
| DELETE User  WHERE id=$id | |

|  |  |
| --- | --- |
| SQL605 | Delete project |
| Web resource | R607 |
| DELETE Project  WHERE id=$id | |

1. **Transactions**

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  | |

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  | |

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  | |

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  | |

|  |  |
| --- | --- |
| T01 | Change project state |
| Isolation level | SERIALIZABLE |
| Justification | When a project is archived (still exists, but can’t be edited) or deleted, the state of all its boards and tasks must also change (to archived or deleted, respectively). |
| BEGIN TRANSACTION  SET TRANSACTION ISOLATION LEVEL SERIALIZABLE  -- Archive project  UPDATE Project  SET project\_state=$states  WHERE id=$id  -- Archive project boards  UPDATE Board  SET board\_state=$states  WHERE id\_project=$id  -- Archive board tasks  UPDATE Task  SET task\_state=$states  INNER JOIN Board ON Board.id=Task.id\_board  WHERE Board.id\_project=$id  COMMIT | |

|  |  |
| --- | --- |
| T02 | Delete user |
| Isolation level | SERIALIZABLE |
| Justification | When a user is deleted (removed from database), all the projects he coordinated are archived, as well as all their boards and projects. |
| BEGIN TRANSACTION  SET TRANSACTION ISOLATION LEVEL SERIALIZABLE  -- Delete user  DELETE FROM User  WHERE id=$userID  -- Archive project  UPDATE Project  SET project\_state='Archived'  WHERE id\_coordinator=$userID  -- Archive project boards  UPDATE Board  SET board\_state='Archived'  INNER JOIN Project ON Board.id\_project=Project.id  WHERE Porject.id\_coordinator=$userID  -- Archive board tasks  UPDATE Task  SET task\_state='Archived'  INNER JOIN Board ON Board.id=Task.id\_board  INNER JOIN Project ON Project.id=Board.id\_project  WHERE Project.id\_coordinator=$userID  COMMIT | |

|  |  |
| --- | --- |
| T03 | Get notifications |
| Isolation level | SERIALIZABLE READ ONLY |
| Justification | In the middle of the transaction, the insertion of new rows in the loan table can occur, which implies that the information retrieved in both selects is different, consequently resulting in a Phantom Read. It's READ ONLY because it only uses Selects. |
| BEGIN TRANSACTION  SET TRANSACTION ISOLATION LEVEL SERIALIZABLE READ ONLY  -- Get number of unread notifications  SELECT COUNT id  FROM Notification  WHERE Notification.id\_user=$user AND Notification.read=0  -- Get notifications  SELECT \* FROM Notification  WHERE Notification.id\_user=$user  COMMIT | |

Changes made to last submission:

* Added more queries to every module, in order to increase complexity and cover all necessary queries
* Dropped previous transactions, that were unnecessary
* Added 3 new transactions

GROUP1734, 21/04/2018:

* Maria Inês Gonçalves, up201402784@fe.up.pt
* Maria Teresa Valério, up201405655@fe.up.pt
* Sara Gomes, up201405085@fe.up.pt