

```

1  --
2  -- PostgreSQL database dump
3  --
4
5  -- Dumped from database version 14.1
6  -- Dumped by pg_dump version 14.3
7
8  SET statement_timeout = 0;
9  SET lock_timeout = 0;
10 SET idle_in_transaction_session_timeout = 0;
11 SET client_encoding = 'UTF8';
12 SET standard_conforming_strings = on;
13 SELECT pg_catalog.set_config('search_path', '', false);
14 SET check_function_bodies = false;
15 SET xmloption = content;
16 SET client_min_messages = warning;
17 SET row_security = off;
18
19 --
20 -- Name: auth; Type: SCHEMA; Schema: -; Owner: supabase_admin
21 --
22
23 CREATE SCHEMA auth;
24
25
26 ALTER SCHEMA auth OWNER TO supabase_admin;
27
28 --
29 -- Name: extensions; Type: SCHEMA; Schema: -; Owner: postgres
30 --
31
32 CREATE SCHEMA extensions;
33
34
35 ALTER SCHEMA extensions OWNER TO postgres;
36
37 --
38 -- Name: pg_graphql; Type: EXTENSION; Schema: -; Owner: -
39 --
40
41 CREATE EXTENSION IF NOT EXISTS pg_graphql WITH SCHEMA public;
42
43
44 --
45 -- Name: EXTENSION pg_graphql; Type: COMMENT; Schema: -; Owner:
46 --
47
48 COMMENT ON EXTENSION pg_graphql IS 'GraphQL support';
49
50
51 --
52 -- Name: graphql_public; Type: SCHEMA; Schema: -; Owner: supabase_admin
53 --
54
55 CREATE SCHEMA graphql_public;
56
57
58 ALTER SCHEMA graphql_public OWNER TO supabase_admin;
59
60 --
61 -- Name: pgbouncer; Type: SCHEMA; Schema: -; Owner: pgbouncer
62 --
63
64 CREATE SCHEMA pgbouncer;

```

```
65
66
67 ALTER SCHEMA pgbouncer OWNER TO pgbouncer;
68
69 --
70 -- Name: realtime; Type: SCHEMA; Schema: -; Owner: supabase_admin
71 --
72
73 CREATE SCHEMA realtime;
74
75
76 ALTER SCHEMA realtime OWNER TO supabase_admin;
77
78 --
79 -- Name: storage; Type: SCHEMA; Schema: -; Owner: supabase_admin
80 --
81
82 CREATE SCHEMA storage;
83
84
85 ALTER SCHEMA storage OWNER TO supabase_admin;
86
87 --
88 -- Name: pg_stat_statements; Type: EXTENSION; Schema: -; Owner: -
89 --
90
91 CREATE EXTENSION IF NOT EXISTS pg_stat_statements WITH SCHEMA extensions;
92
93
94 --
95 -- Name: EXTENSION pg_stat_statements; Type: COMMENT; Schema: -; Owner:
96 --
97
98 COMMENT ON EXTENSION pg_stat_statements IS 'track planning and execution statistics of
99 all SQL statements executed';
100
101 --
102 -- Name: pgcrypto; Type: EXTENSION; Schema: -; Owner: -
103 --
104
105 CREATE EXTENSION IF NOT EXISTS pgcrypto WITH SCHEMA extensions;
106
107
108 --
109 -- Name: EXTENSION pgcrypto; Type: COMMENT; Schema: -; Owner:
110 --
111
112 COMMENT ON EXTENSION pgcrypto IS 'cryptographic functions';
113
114
115 --
116 -- Name: pgjwt; Type: EXTENSION; Schema: -; Owner: -
117 --
118
119 CREATE EXTENSION IF NOT EXISTS pgjwt WITH SCHEMA extensions;
120
121
122 --
123 -- Name: EXTENSION pgjwt; Type: COMMENT; Schema: -; Owner:
124 --
125
126 COMMENT ON EXTENSION pgjwt IS 'JSON Web Token API for Postgresql';
127
```

```
128
129 --
130 -- Name: uuid-oss; Type: EXTENSION; Schema: -; Owner: -
131 --
132
133 CREATE EXTENSION IF NOT EXISTS "uuid-oss" WITH SCHEMA extensions;
134
135
136 --
137 -- Name: EXTENSION "uuid-oss"; Type: COMMENT; Schema: -; Owner:
138 --
139
140 COMMENT ON EXTENSION "uuid-oss" IS 'generate universally unique identifiers (UUIDs)';
141
142
143 --
144 -- Name: action; Type: TYPE; Schema: realtime; Owner: supabase_admin
145 --
146
147 CREATE TYPE realtime.action AS ENUM (
148     'INSERT',
149     'UPDATE',
150     'DELETE',
151     'TRUNCATE',
152     'ERROR'
153 );
154
155
156 ALTER TYPE realtime.action OWNER TO supabase_admin;
157
158 --
159 -- Name: equality_op; Type: TYPE; Schema: realtime; Owner: supabase_admin
160 --
161
162 CREATE TYPE realtime.equality_op AS ENUM (
163     'eq',
164     'neq',
165     'lt',
166     'lte',
167     'gt',
168     'gte'
169 );
170
171
172 ALTER TYPE realtime.equality_op OWNER TO supabase_admin;
173
174 --
175 -- Name: user_defined_filter; Type: TYPE; Schema: realtime; Owner: supabase_admin
176 --
177
178 CREATE TYPE realtime.user_defined_filter AS (
179     column_name text,
180     op realtime.equality_op,
181     value text
182 );
183
184
185 ALTER TYPE realtime.user_defined_filter OWNER TO supabase_admin;
186
187 --
188 -- Name: wal_column; Type: TYPE; Schema: realtime; Owner: supabase_admin
189 --
190
191 CREATE TYPE realtime.wal_column AS (
```

```

192     name text,
193     type text,
194     value jsonb,
195     is_pkey boolean,
196     is_selectable boolean
197 );
198
199
200 ALTER TYPE realtime.wal_column OWNER TO supabase_admin;
201
202 --
203 -- Name: wal_rls; Type: TYPE; Schema: realtime; Owner: supabase_admin
204 --
205
206 CREATE TYPE realtime.wal_rls AS (
207     wal jsonb,
208     is_rls_enabled boolean,
209     subscription_ids uuid[],
210     errors text[]
211 );
212
213
214 ALTER TYPE realtime.wal_rls OWNER TO supabase_admin;
215
216 --
217 -- Name: email(); Type: FUNCTION; Schema: auth; Owner: supabase_auth_admin
218 --
219
220 CREATE FUNCTION auth.email() RETURNS text
221     LANGUAGE sql STABLE
222     AS $$
223     select
224         coalesce(
225             nullif(current_setting('request.jwt.claim.email', true), ''),
226             (nullif(current_setting('request.jwt.claims', true), '')::jsonb ->> 'email')
227         )::text
228 $$;
229
230
231 ALTER FUNCTION auth.email() OWNER TO supabase_auth_admin;
232
233 --
234 -- Name: role(); Type: FUNCTION; Schema: auth; Owner: supabase_auth_admin
235 --
236
237 CREATE FUNCTION auth.role() RETURNS text
238     LANGUAGE sql STABLE
239     AS $$
240     select
241         coalesce(
242             nullif(current_setting('request.jwt.claim.role', true), ''),
243             (nullif(current_setting('request.jwt.claims', true), '')::jsonb ->> 'role')
244         )::text
245 $$;
246
247
248 ALTER FUNCTION auth.role() OWNER TO supabase_auth_admin;
249
250 --
251 -- Name: uid(); Type: FUNCTION; Schema: auth; Owner: supabase_auth_admin
252 --
253
254 CREATE FUNCTION auth.uid() RETURNS uuid
255     LANGUAGE sql STABLE

```

```

256     AS $$
257 select
258     coalesce(
259         nullif(current_setting('request.jwt.claim.sub', true), ''),
260         (nullif(current_setting('request.jwt.claims', true), '')::jsonb ->> 'sub')
261     )::uuid
262 $$;
263
264
265 ALTER FUNCTION auth.uid() OWNER TO supabase_auth_admin;
266
267 --
268 -- Name: grant_pg_cron_access(); Type: FUNCTION; Schema: extensions; Owner: postgres
269 --
270
271 CREATE FUNCTION extensions.grant_pg_cron_access() RETURNS event_trigger
272     LANGUAGE plpgsql
273     AS $$
274 DECLARE
275     schema_is_cron bool;
276 BEGIN
277     schema_is_cron = (
278         SELECT n.nspname = 'cron'
279         FROM pg_event_trigger_ddl_commands() AS ev
280         LEFT JOIN pg_catalog.pg_namespace AS n
281             ON ev.objid = n.oid
282     );
283
284 IF schema_is_cron
285 THEN
286     grant usage on schema cron to postgres with grant option;
287
288     alter default privileges in schema cron grant all on tables to postgres with grant
289     option;
290     alter default privileges in schema cron grant all on functions to postgres with
291     grant option;
292     alter default privileges in schema cron grant all on sequences to postgres with
293     grant option;
294
295     alter default privileges for user supabase_admin in schema cron grant all
296     on sequences to postgres with grant option;
297     alter default privileges for user supabase_admin in schema cron grant all
298     on tables to postgres with grant option;
299     alter default privileges for user supabase_admin in schema cron grant all
300     on functions to postgres with grant option;
301
302     grant all privileges on all tables in schema cron to postgres with grant option;
303
304 END IF;
305
306 END;
307
308 $$;
309
310
311 ALTER FUNCTION extensions.grant_pg_cron_access() OWNER TO postgres;
312
313 --
314 -- Name: FUNCTION grant_pg_cron_access(); Type: COMMENT; Schema: extensions; Owner:
315 postgres
316 --
317
318 COMMENT ON FUNCTION extensions.grant_pg_cron_access() IS 'Grants access to pg_cron';
319
320

```

```

316 --
317 -- Name: grant_pg_graphql_access(); Type: FUNCTION; Schema: extensions; Owner:
supabase_admin
318 --
319
320 CREATE FUNCTION extensions.grant_pg_graphql_access() RETURNS event_trigger
321     LANGUAGE plpgsql
322     AS $_$
323 DECLARE
324     func_is_graphql_resolve bool;
325 BEGIN
326     func_is_graphql_resolve = (
327         SELECT n.proname = 'resolve'
328         FROM pg_event_trigger_ddl_commands() AS ev
329         LEFT JOIN pg_catalog.pg_proc AS n
330         ON ev.objid = n.oid
331     );
332
333     IF func_is_graphql_resolve
334     THEN
335         grant usage on schema graphql to postgres, anon, authenticated, service_role;
336         grant all on function graphql.resolve to postgres, anon, authenticated,
service_role;
337
338         alter default privileges in schema graphql grant all on tables to postgres, anon
, authenticated, service_role;
339         alter default privileges in schema graphql grant all on functions to postgres,
anon, authenticated, service_role;
340         alter default privileges in schema graphql grant all on sequences to postgres,
anon, authenticated, service_role;
341
342         -- Update public wrapper to pass all arguments through to the pg_graphql
resolve func
343         create or replace function graphql_public.graphql(
344             "operationName" text default null,
345             query text default null,
346             variables jsonb default null,
347             extensions jsonb default null
348         )
349             returns jsonb
350             language sql
351         as $$
352             -- This changed
353             select graphql.resolve(
354                 query := query,
355                 variables := coalesce(variables, '{}'),
356                 "operationName" := "operationName",
357                 extensions := extensions
358             );
359         $$;
360
361         grant select on graphql.field, graphql.type, graphql.enum_value to postgres,
anon, authenticated, service_role;
362         grant execute on function graphql.resolve to postgres, anon, authenticated,
service_role;
363     END IF;
364
365 END;
366 $_$;
367
368
369 ALTER FUNCTION extensions.grant_pg_graphql_access() OWNER TO supabase_admin;
370
371 --

```

```

372 -- Name: FUNCTION grant_pg_graphql_access(); Type: COMMENT; Schema: extensions; Owner:
supabase_admin
373 --
374
375 COMMENT ON FUNCTION extensions.grant_pg_graphql_access() IS 'Grants access to
pg_graphql';
376
377
378 --
379 -- Name: grant_pg_net_access(); Type: FUNCTION; Schema: extensions; Owner: postgres
380 --
381
382 CREATE FUNCTION extensions.grant_pg_net_access() RETURNS event_trigger
383     LANGUAGE plpgsql
384     AS $$
385 BEGIN
386     IF EXISTS (
387         SELECT 1
388         FROM pg_event_trigger_ddl_commands() AS ev
389         JOIN pg_extension AS ext
390         ON ev.objid = ext.oid
391         WHERE ext.extname = 'pg_net'
392     )
393 THEN
394     IF NOT EXISTS (
395         SELECT 1
396         FROM pg_roles
397         WHERE rolname = 'supabase_functions_admin'
398     )
399 THEN
400     CREATE USER supabase_functions_admin NOINHERIT CREATEROLE LOGIN NOREPLICATION;
401 END IF;
402
403 GRANT USAGE ON SCHEMA net TO supabase_functions_admin, postgres, anon, authenticated
, service_role;
404
405 ALTER function net.http_get(url text, params jsonb, headers jsonb,
timeout_milliseconds integer) SECURITY DEFINER;
406 ALTER function net.http_post(url text, body jsonb, params jsonb, headers jsonb,
timeout_milliseconds integer) SECURITY DEFINER;
407 ALTER function net.http_collect_response(request_id bigint, async boolean) SECURITY
DEFINER;
408
409 ALTER function net.http_get(url text, params jsonb, headers jsonb,
timeout_milliseconds integer) SET search_path = net;
410 ALTER function net.http_post(url text, body jsonb, params jsonb, headers jsonb,
timeout_milliseconds integer) SET search_path = net;
411 ALTER function net.http_collect_response(request_id bigint, async boolean) SET
search_path = net;
412
413 REVOKE ALL ON FUNCTION net.http_get(url text, params jsonb, headers jsonb,
timeout_milliseconds integer) FROM PUBLIC;
414 REVOKE ALL ON FUNCTION net.http_post(url text, body jsonb, params jsonb, headers
jsonb, timeout_milliseconds integer) FROM PUBLIC;
415 REVOKE ALL ON FUNCTION net.http_collect_response(request_id bigint, async boolean)
FROM PUBLIC;
416
417 GRANT EXECUTE ON FUNCTION net.http_get(url text, params jsonb, headers jsonb,
timeout_milliseconds integer) TO supabase_functions_admin, postgres, anon,
authenticated, service_role;
418 GRANT EXECUTE ON FUNCTION net.http_post(url text, body jsonb, params jsonb, headers
jsonb, timeout_milliseconds integer) TO supabase_functions_admin, postgres, anon,
authenticated, service_role;
419 GRANT EXECUTE ON FUNCTION net.http_collect_response(request_id bigint, async boolean

```

```

    ) TO supabase_functions_admin, postgres, anon, authenticated, service_role;
420 END IF;
421 END;
422 $$;
423
424
425 ALTER FUNCTION extensions.grant_pg_net_access() OWNER TO postgres;
426
427 --
428 -- Name: FUNCTION grant_pg_net_access(); Type: COMMENT; Schema: extensions; Owner:
postgres
429 --
430
431 COMMENT ON FUNCTION extensions.grant_pg_net_access() IS 'Grants access to pg_net';
432
433 --
434 -- Name: pgrst_ddl_watch(); Type: FUNCTION; Schema: extensions; Owner: supabase_admin
435 --
436
437 CREATE FUNCTION extensions.pgrst_ddl_watch() RETURNS event_trigger
438     LANGUAGE plpgsql
439     AS $$
440 DECLARE
441     cmd record;
442 BEGIN
443     FOR cmd IN SELECT * FROM pg_event_trigger_ddl_commands()
444     LOOP
445         IF cmd.command_tag IN (
446             'CREATE SCHEMA', 'ALTER SCHEMA'
447             , 'CREATE TABLE', 'CREATE TABLE AS', 'SELECT INTO', 'ALTER TABLE'
448             , 'CREATE FOREIGN TABLE', 'ALTER FOREIGN TABLE'
449             , 'CREATE VIEW', 'ALTER VIEW'
450             , 'CREATE MATERIALIZED VIEW', 'ALTER MATERIALIZED VIEW'
451             , 'CREATE FUNCTION', 'ALTER FUNCTION'
452             , 'CREATE TRIGGER'
453             , 'CREATE TYPE', 'ALTER TYPE'
454             , 'CREATE RULE'
455             , 'COMMENT'
456         )
457         -- don't notify in case of CREATE TEMP table or other objects created on pg_temp
458         AND cmd.schema_name is distinct from 'pg_temp'
459         THEN
460             NOTIFY pgrst, 'reload schema';
461         END IF;
462     END LOOP;
463 END; $$;
464
465
466 ALTER FUNCTION extensions.pgrst_ddl_watch() OWNER TO supabase_admin;
467
468 --
469 -- Name: pgrst_drop_watch(); Type: FUNCTION; Schema: extensions; Owner: supabase_admin
470 --
471
472 CREATE FUNCTION extensions.pgrst_drop_watch() RETURNS event_trigger
473     LANGUAGE plpgsql
474     AS $$
475 DECLARE
476     obj record;
477 BEGIN
478     FOR obj IN SELECT * FROM pg_event_trigger_dropped_objects()
479     LOOP
480         IF obj.object_type IN (

```



```

482         'schema'
483     , 'table'
484     , 'foreign table'
485     , 'view'
486     , 'materialized view'
487     , 'function'
488     , 'trigger'
489     , 'type'
490     , 'rule'
491 )
492 AND obj.is_temporary IS false -- no pg_temp objects
493 THEN
494     NOTIFY pgrst, 'reload schema';
495 END IF;
496 END LOOP;
497 END; $$;
498
499
500 ALTER FUNCTION extensions.pgrst_drop_watch() OWNER TO supabase_admin;
501
502 --
503 -- Name: set_graphql_placeholder(); Type: FUNCTION; Schema: extensions; Owner:
supabase_admin
504 --
505
506 CREATE FUNCTION extensions.set_graphql_placeholder() RETURNS event_trigger
507     LANGUAGE plpgsql
508     AS $_$
509     DECLARE
510         graphql_is_dropped bool;
511     BEGIN
512         graphql_is_dropped = (
513             SELECT ev.schema_name = 'graphql_public'
514             FROM pg_event_trigger_dropped_objects() AS ev
515             WHERE ev.schema_name = 'graphql_public'
516         );
517
518         IF graphql_is_dropped
519         THEN
520             create or replace function graphql_public.graphql(
521                 "operationName" text default null,
522                 query text default null,
523                 variables jsonb default null,
524                 extensions jsonb default null
525             )
526                 returns jsonb
527                 language plpgsql
528             as $$
529                 DECLARE
530                     server_version float;
531                 BEGIN
532                     server_version = (SELECT (SPLIT_PART((select version()), ' ', 2))::float
533                                     );
534
535                     IF server_version >= 14 THEN
536                         RETURN jsonb_build_object(
537                             'errors', jsonb_build_array(
538                                 jsonb_build_object(
539                                     'message', 'pg_graphql extension is not enabled.'
540                                 )
541                             )
542                         );
543                     ELSE
544                         RETURN jsonb_build_object(

```

```

544         'errors', jsonb_build_array(
545             jsonb_build_object(
546                 'message', 'pg_graphql is only available on projects
                    running Postgres 14 onwards.'
547             )
548         )
549     );
550     END IF;
551 END;
552 $$;
553 END IF;
554
555 END;
556 $_$;
557
558
559 ALTER FUNCTION extensions.set_graphql_placeholder() OWNER TO supabase_admin;
560
561 --
562 -- Name: FUNCTION set_graphql_placeholder(); Type: COMMENT; Schema: extensions; Owner:
supabase_admin
563 --
564
565 COMMENT ON FUNCTION extensions.set_graphql_placeholder() IS 'Reintroduces placeholder
function for graphql_public.graphql';
566
567
568 --
569 -- Name: get_auth(text); Type: FUNCTION; Schema: pgbouncer; Owner: postgres
570 --
571
572 CREATE FUNCTION pgbouncer.get_auth(p_username text) RETURNS TABLE(username text, password
text)
573     LANGUAGE plpgsql SECURITY DEFINER
574     AS $$
575 BEGIN
576     RAISE WARNING 'PgBouncer auth request: %', p_username;
577
578     RETURN QUERY
579     SELECT username::TEXT, passwd::TEXT FROM pg_catalog.pg_shadow
580     WHERE username = p_username;
581 END;
582 $$;
583
584
585 ALTER FUNCTION pgbouncer.get_auth(p_username text) OWNER TO postgres;
586
587 --
588 -- Name: apply_qls(jsonb, integer); Type: FUNCTION; Schema: realtime; Owner:
supabase_admin
589 --
590
591 CREATE FUNCTION realtime.apply_qls(wal jsonb, max_record_bytes integer DEFAULT (1024 *
1024)) RETURNS SETOF realtime.wal_qls
592     LANGUAGE plpgsql
593     AS $$
594     declare
595         -- Regclass of the table e.g. public.notes
596         entity_regclass = (quote_ident(wal ->> 'schema') || '.' || quote_ident(wal ->>
'table'))::regclass;
597
598         -- I, U, D, T: insert, update ...
599         action realtime.action = (
600             case wal ->> 'action'

```

```

601         when 'I' then 'INSERT'
602         when 'U' then 'UPDATE'
603         when 'D' then 'DELETE'
604         else 'ERROR'
605     end
606 );
607
608 -- Is row level security enabled for the table
609 is_rls_enabled bool = relrowsecurity from pg_class where oid = entity_;
610
611 subscriptions realtime.subscription[] = array_agg(subs)
612     from
613         realtime.subscription subs
614     where
615         subs.entity = entity_;
616
617 -- Subscription vars
618 roles regrole[] = array_agg(distinct us.claims_role)
619     from
620         unnest(subscriptions) us;
621
622 working_role regrole;
623 claimed_role regrole;
624 claims jsonb;
625
626 subscription_id uuid;
627 subscription_has_access bool;
628 visible_to_subscription_ids uuid[] = '{}';
629
630 -- structured info for wal's columns
631 columns realtime.wal_column[];
632 -- previous identity values for update/delete
633 old_columns realtime.wal_column[];
634
635 error_record_exceeds_max_size boolean = octet_length(wal::text) >
max_record_bytes;
636
637 -- Primary jsonb output for record
638 output jsonb;
639
640 begin
641     perform set_config('role', null, true);
642
643     columns =
644         array_agg(
645             (
646                 x->>'name',
647                 x->>'type',
648                 realtime.cast((x->>'value') #>> '{}', (x->>'type')::regtype),
649                 (pks ->> 'name') is not null,
650                 true
651             )::realtime.wal_column
652         )
653     from
654         jsonb_array_elements(wal -> 'columns') x
655     left join jsonb_array_elements(wal -> 'pk') pks
656         on (x ->> 'name') = (pks ->> 'name');
657
658     old_columns =
659         array_agg(
660             (
661                 x->>'name',
662                 x->>'type',
663                 realtime.cast((x->>'value') #>> '{}', (x->>'type')::regtype),

```

```

664         (pks ->> 'name') is not null,
665         true
666     )::realtime.wal_column
667 )
668 from
669     jsonb_array_elements(wal -> 'identity') x
670     left join jsonb_array_elements(wal -> 'pk') pks
671         on (x ->> 'name') = (pks ->> 'name');
672
673 for working_role in select * from unnest(roles) loop
674
675     -- Update `is_selectable` for columns and old_columns
676     columns =
677         array_agg(
678             (
679                 c.name,
680                 c.type,
681                 c.value,
682                 c.is_pkey,
683                 pg_catalog.has_column_privilege(working_role, entity_, c.name, 'SELECT')
684             )::realtime.wal_column
685         )
686     from
687         unnest(columns) c;
688
689     old_columns =
690         array_agg(
691             (
692                 c.name,
693                 c.type,
694                 c.value,
695                 c.is_pkey,
696                 pg_catalog.has_column_privilege(working_role, entity_, c.name, 'SELECT')
697             )::realtime.wal_column
698         )
699     from
700         unnest(old_columns) c;
701
702     if action <> 'DELETE' and count(1) = 0 from unnest(columns) c where c.is_pkey
703     then
704         return next (
705             null,
706             is_rls_enabled,
707             -- subscriptions is already filtered by entity
708             (select array_agg(s.subscription_id) from unnest(subscriptions) as s where
709                 claims_role = working_role),
710             array['Error 400: Bad Request, no primary key']
711         )::realtime.wal_rls;
712
713     -- The claims role does not have SELECT permission to the primary key of entity
714     elsif action <> 'DELETE' and sum(c.is_selectable::int) <> count(1) from unnest
715     (columns) c where c.is_pkey then
716         return next (
717             null,
718             is_rls_enabled,
719             (select array_agg(s.subscription_id) from unnest(subscriptions) as s where
720                 claims_role = working_role),
721             array['Error 401: Unauthorized']
722         )::realtime.wal_rls;
723
724     else
725         output = jsonb_build_object(
726             'schema', wal ->> 'schema',
727             'table', wal ->> 'table',

```

```

724         'type', action,
725         'commit_timestamp', to_char(
726             (wal ->> 'timestamp')::timestamp,
727             'YYYY-MM-DD"T"HH24:MI:SS"Z" '
728     ),
729     'columns', (
730         select
731             jsonb_agg(
732                 jsonb_build_object(
733                     'name', pa.attname,
734                     'type', pt.typname
735                 )
736             order by pa.attnum asc
737         )
738         from
739             pg_attribute pa
740             join pg_type pt
741                 on pa.atttypid = pt.oid
742         where
743             attrelid = entity_
744             and attnum > 0
745             and pg_catalog.has_column_privilege(working_role, entity_, pa.
746                 attname, 'SELECT')
747     )
748 -- Add "record" key for insert and update
749 || case
750     when error_record_exceeds_max_size then jsonb_build_object('record',
751         '{}'::jsonb)
752     when action in ('INSERT', 'UPDATE') then
753         jsonb_build_object(
754             'record',
755             (select jsonb_object_agg((c).name, (c).value) from unnest(columns) c
756              where (c).is_selectable)
757         )
758     else '{}'::jsonb
759 end
760 -- Add "old_record" key for update and delete
761 || case
762     when error_record_exceeds_max_size then jsonb_build_object('old_record',
763         '{}'::jsonb)
764     when action in ('UPDATE', 'DELETE') then
765         jsonb_build_object(
766             'old_record',
767             (select jsonb_object_agg((c).name, (c).value) from unnest(
768                 old_columns) c where (c).is_selectable)
769         )
770     else '{}'::jsonb
771 end;
772
773 -- Create the prepared statement
774 if is_ri_enabled and action <> 'DELETE' then
775     if (select 1 from pg_prepared_statements where name = 'walrus_ri_stmt'
776         limit 1) > 0 then
777         deallocate walrus_ri_stmt;
778     end if;
779     execute realtime.build_prepared_statement_sql('walrus_ri_stmt', entity_,
780         columns);
781 end if;
782
783 visible_to_subscription_ids = '{}';
784
785 for subscription_id, claims in (
786     select

```

```

781         subs.subscription_id,
782         subs.claims
783     from
784         unnest(subscriptions) subs
785     where
786         subs.entity = entity_
787         and subs.claims_role = working_role
788         and realtime.is_visible_through_filters(columns, subs.filters)
789 ) loop
790
791 if not is_rls_enabled or action = 'DELETE' then
792     visible_to_subscription_ids = visible_to_subscription_ids ||
793     subscription_id;
794 else
795     -- Check if RLS allows the role to see the record
796     perform
797         set_config('role', working_role::text, true),
798         set_config('request.jwt.claims', claims::text, true);
799
800     execute 'execute walrus_rls_stmt' into subscription_has_access;
801
802     if subscription_has_access then
803         visible_to_subscription_ids = visible_to_subscription_ids ||
804         subscription_id;
805     end if;
806 end if;
807 end loop;
808
809 perform set_config('role', null, true);
810
811 return next (
812     output,
813     is_rls_enabled,
814     visible_to_subscription_ids,
815     case
816         when error_record_exceeds_max_size then array['Error 413: Payload Too
817         Large']
818         else '{}'
819     end
820 ):realtime.wal_rls;
821
822 end if;
823 end loop;
824
825 perform set_config('role', null, true);
826 end;
827 $$;
828
829 ALTER FUNCTION realtime.apply_rls(wal jsonb, max_record_bytes integer) OWNER TO
830 supabase_admin;
831
832 --
833 -- Name: build_prepared_statement_sql(text, regclass, realtime.wal_column[]); Type:
834 FUNCTION; Schema: realtime; Owner: supabase_admin
835 --
836
837 CREATE FUNCTION realtime.build_prepared_statement_sql(prepared_statement_name text,
838 entity regclass, columns realtime.wal_column[]) RETURNS text
839 LANGUAGE sql
840 AS $$
841 /*
842 Builds a sql string that, if executed, creates a prepared statement to
843 tests retrieve a row from *entity* by its primary key columns.

```

```

839
840 Example
841 select realtime.build_prepared_statement_sql('public.notes', '{"id"}'::text[],
842 '{"bigint"}'::text[])
843 */
844 select
845 'prepare ' || prepared_statement_name || ' as
846 select
847     exists(
848         select
849             1
850         from
851             ' || entity || '
852         where
853             ' || string_agg(quote_ident(pkc.name) || '=' || quote_nullable(pkc.value #>>
854                 '{}') , ' and ') || '
855     )'
856 from
857     unnest(columns) pkc
858 where
859     pkc.is_pkey
860 group by
861     entity
862 $$;
863
864 ALTER FUNCTION realtime.build_prepared_statement_sql(prepared_statement_name text,
865 entity regclass, columns realtime.wal_column[]) OWNER TO supabase_admin;
866
867 --
868 -- Name: cast(text, regtype); Type: FUNCTION; Schema: realtime; Owner: supabase_admin
869
870 CREATE FUNCTION realtime."cast"(val text, type_ regtype) RETURNS jsonb
871 LANGUAGE plpgsql IMMUTABLE
872 AS $$
873 declare
874     res jsonb;
875 begin
876     execute format('select to_jsonb(%L::' || type_::text || ')', val) into res;
877     return res;
878 end
879 $$;
880
881 ALTER FUNCTION realtime."cast"(val text, type_ regtype) OWNER TO supabase_admin;
882
883 --
884 -- Name: check_equality_op(realtime.equality_op, regtype, text, text); Type: FUNCTION;
885 Schema: realtime; Owner: supabase_admin
886
887 --
888 CREATE FUNCTION realtime.check_equality_op(op realtime.equality_op, type_ regtype, val_1
889 text, val_2 text) RETURNS boolean
890 LANGUAGE plpgsql IMMUTABLE
891 AS $$
892 /*
893 Casts *val_1* and *val_2* as type *type_* and check the *op* condition for truthiness
894 */
895 declare
896     op_symbol text = (
897         case
898             when op = 'eq' then '='
899             when op = 'neq' then '!='

```

```

898         when op = 'lt' then '<'
899         when op = 'lte' then '<='
900         when op = 'gt' then '>'
901         when op = 'gte' then '>='
902         else 'UNKNOWN OP'
903     end
904 );
905 res boolean;
906 begin
907     execute format('select %L::' || type_::text || ' ' || op_symbol || ' ' || %L::' || type_
908         ::text, val_1, val_2) into res;
909     return res;
910 end;
911 $$;
912
913 ALTER FUNCTION realtime.check_equality_op(op realtime.equality_op, type_ regtype, val_1
914 text, val_2 text) OWNER TO supabase_admin;
915
916 --
917 -- Name: is_visible_through_filters(realtime.wal_column[],
918 realtime.user_defined_filter[]); Type: FUNCTION; Schema: realtime; Owner: supabase_admin
919
920 CREATE FUNCTION realtime.is_visible_through_filters(columns realtime.wal_column[],
921 filters realtime.user_defined_filter[]) RETURNS boolean
922 LANGUAGE sql IMMUTABLE
923 AS $$
924 /*
925 Should the record be visible (true) or filtered out (false) after *filters* are
926 applied
927 */
928 select
929     -- Default to allowed when no filters present
930     coalesce(
931         sum(
932             realtime.check_equality_op(
933                 op:=f.op,
934                 type_:=col.type::regtype,
935                 -- cast jsonb to text
936                 val_1:=col.value #>> '{}',
937                 val_2:=f.value
938             )::int
939         ) = count(1),
940         true
941     )
942 from
943     unnest(filters) f
944     join unnest(columns) col
945     on f.column_name = col.name;
946 $$;
947
948 ALTER FUNCTION realtime.is_visible_through_filters(columns realtime.wal_column[],
949 filters realtime.user_defined_filter[]) OWNER TO supabase_admin;
950
951 --
952 -- Name: quote_wal2json(regclass); Type: FUNCTION; Schema: realtime; Owner:
953 supabase_admin
954
955 CREATE FUNCTION realtime.quote_wal2json(entity regclass) RETURNS text
956 LANGUAGE sql IMMUTABLE STRICT
957 AS $$

```



```

955     select
956     (
957         select string_agg(' ' || ch, '')
958         from unnest(string_to_array(nsp.nspname::text, null)) with ordinality x(ch,
959         idx)
960         where
961             not (x.idx = 1 and x.ch = '')
962             and not (
963                 x.idx = array_length(string_to_array(nsp.nspname::text, null), 1)
964                 and x.ch = ''
965             )
966         || ' '
967         || (
968             select string_agg(' ' || ch, '')
969             from unnest(string_to_array(pc.relname::text, null)) with ordinality x(ch, idx)
970             where
971                 not (x.idx = 1 and x.ch = '')
972                 and not (
973                     x.idx = array_length(string_to_array(nsp.nspname::text, null), 1)
974                     and x.ch = ''
975                 )
976             )
977         from
978             pg_class pc
979             join pg_namespace nsp
980             on pc.relnamespace = nsp.oid
981         where
982             pc.oid = entity
983     $$;
984
985
986 ALTER FUNCTION realtime.quote_wal2json(entity regclass) OWNER TO supabase_admin;
987
988 --
989 -- Name: subscription_check_filters(); Type: FUNCTION; Schema: realtime; Owner:
990 -- supabase_admin
991
992 CREATE FUNCTION realtime.subscription_check_filters() RETURNS trigger
993     LANGUAGE plpgsql
994     AS $$
995     /*
996     Validates that the user defined filters for a subscription:
997     - refer to valid columns that the claimed role may access
998     - values are coercable to the correct column type
999     */
1000     declare
1001         col_names text[] = coalesce(
1002             array_agg(c.column_name order by c.ordinal_position),
1003             '{}'::text[]
1004         )
1005     from
1006         information_schema.columns c
1007     where
1008         format('%I.%I', c.table_schema, c.table_name)::regclass = new.entity
1009         and pg_catalog.has_column_privilege(
1010             (new.claims ->> 'role'),
1011             format('%I.%I', c.table_schema, c.table_name)::regclass,
1012             c.column_name,
1013             'SELECT'
1014         );
1015     filter realtime.user_defined_filter;
1016     col_type regtype;

```

```

1017 begin
1018     for filter in select * from unnest(new.filters) loop
1019         -- Filtered column is valid
1020         if not filter.column_name = any(col_names) then
1021             raise exception 'invalid column for filter %', filter.column_name;
1022         end if;
1023
1024         -- Type is sanitized and safe for string interpolation
1025         col_type = (
1026             select atttypid::regtype
1027             from pg_catalog.pg_attribute
1028             where attrelid = new.entity
1029                 and attname = filter.column_name
1030         );
1031         if col_type is null then
1032             raise exception 'failed to lookup type for column %', filter.column_name;
1033         end if;
1034         -- raises an exception if value is not coercable to type
1035         perform realtime.cast(filter.value, col_type);
1036     end loop;
1037
1038     -- Apply consistent order to filters so the unique constraint on
1039     -- (subscription_id, entity, filters) can't be tricked by a different filter order
1040     new.filters = coalesce(
1041         array_agg(f order by f.column_name, f.op, f.value),
1042         '{}'
1043     ) from unnest(new.filters) f;
1044
1045     return new;
1046 end;
1047 $$;
1048
1049
1050 ALTER FUNCTION realtime.subscription_check_filters() OWNER TO supabase_admin;
1051
1052 --
1053 -- Name: to_regrole(text); Type: FUNCTION; Schema: realtime; Owner: supabase_admin
1054 --
1055
1056 CREATE FUNCTION realtime.to_regrole(role_name text) RETURNS regrole
1057     LANGUAGE sql IMMUTABLE
1058     AS $$ select role_name::regrole $$;
1059
1060
1061 ALTER FUNCTION realtime.to_regrole(role_name text) OWNER TO supabase_admin;
1062
1063 --
1064 -- Name: extension(text); Type: FUNCTION; Schema: storage; Owner: supabase_storage_admin
1065 --
1066
1067 CREATE FUNCTION storage.extension(name text) RETURNS text
1068     LANGUAGE plpgsql
1069     AS $$
1070 DECLARE
1071     _parts text[];
1072     _filename text;
1073 BEGIN
1074     select string_to_array(name, '/') into _parts;
1075     select _parts[array_length(_parts,1)] into _filename;
1076     -- @todo return the last part instead of 2
1077     return split_part(_filename, '.', 2);
1078 END
1079 $$;
1080

```

```

1081
1082 ALTER FUNCTION storage.extension(name text) OWNER TO supabase_storage_admin;
1083
1084 --
1085 -- Name: filename(text); Type: FUNCTION; Schema: storage; Owner: supabase_storage_admin
1086 --
1087
1088 CREATE FUNCTION storage.filename(name text) RETURNS text
1089     LANGUAGE plpgsql
1090     AS $$
1091 DECLARE
1092     _parts text[];
1093 BEGIN
1094     select string_to_array(name, '/') into _parts;
1095     return _parts[array_length(_parts,1)];
1096 END
1097 $$;
1098
1099
1100 ALTER FUNCTION storage.filename(name text) OWNER TO supabase_storage_admin;
1101
1102 --
1103 -- Name: foldername(text); Type: FUNCTION; Schema: storage; Owner: supabase_storage_admin
1104 --
1105
1106 CREATE FUNCTION storage.foldername(name text) RETURNS text[]
1107     LANGUAGE plpgsql
1108     AS $$
1109 DECLARE
1110     _parts text[];
1111 BEGIN
1112     select string_to_array(name, '/') into _parts;
1113     return _parts[1:array_length(_parts,1)-1];
1114 END
1115 $$;
1116
1117
1118 ALTER FUNCTION storage.foldername(name text) OWNER TO supabase_storage_admin;
1119
1120 --
1121 -- Name: get_size_by_bucket(); Type: FUNCTION; Schema: storage; Owner:
supabase_storage_admin
1122 --
1123
1124 CREATE FUNCTION storage.get_size_by_bucket() RETURNS TABLE(size bigint, bucket_id text)
1125     LANGUAGE plpgsql
1126     AS $$
1127 BEGIN
1128     return query
1129         select sum((metadata->>'size')::int) as size, obj.bucket_id
1130         from "storage".objects as obj
1131         group by obj.bucket_id;
1132 END
1133 $$;
1134
1135
1136 ALTER FUNCTION storage.get_size_by_bucket() OWNER TO supabase_storage_admin;
1137
1138 --
1139 -- Name: search(text, text, integer, integer, integer, text, text, text); Type:
FUNCTION; Schema: storage; Owner: supabase_storage_admin
1140 --
1141
1142 CREATE FUNCTION storage.search(prefix text, bucketname text, limits integer DEFAULT 100,

```

```

levels integer DEFAULT 1, offsets integer DEFAULT 0, search text DEFAULT '::text,
sortcolumn text DEFAULT 'name'::text, sortorder text DEFAULT 'asc'::text) RETURNS TABLE(
name text, id uuid, updated_at timestamp with time zone, created_at timestamp with time
zone, last_accessed_at timestamp with time zone, metadata jsonb)
1143     LANGUAGE plpgsql STABLE
1144     AS $_$
1145 declare
1146     v_order_by text;
1147     v_sort_order text;
1148 begin
1149     case
1150         when sortcolumn = 'name' then
1151             v_order_by = 'name';
1152         when sortcolumn = 'updated_at' then
1153             v_order_by = 'updated_at';
1154         when sortcolumn = 'created_at' then
1155             v_order_by = 'created_at';
1156         when sortcolumn = 'last_accessed_at' then
1157             v_order_by = 'last_accessed_at';
1158         else
1159             v_order_by = 'name';
1160     end case;
1161
1162     case
1163         when sortorder = 'asc' then
1164             v_sort_order = 'asc';
1165         when sortorder = 'desc' then
1166             v_sort_order = 'desc';
1167         else
1168             v_sort_order = 'asc';
1169     end case;
1170
1171     v_order_by = v_order_by || ' ' || v_sort_order;
1172
1173     return query execute
1174         'with folders as (
1175             select path_tokens[$1] as folder
1176             from storage.objects
1177             where objects.name ilike $2 || $3 || '%'
1178             and bucket_id = $4
1179             and array_length(regexp_split_to_array(objects.name, '/'), 1) <> $1
1180             group by folder
1181             order by folder ' || v_sort_order || '
1182         )
1183         (select folder as "name",
1184             null as id,
1185             null as updated_at,
1186             null as created_at,
1187             null as last_accessed_at,
1188             null as metadata from folders)
1189     union all
1190     (select path_tokens[$1] as "name",
1191         id,
1192         updated_at,
1193         created_at,
1194         last_accessed_at,
1195         metadata
1196     from storage.objects
1197     where objects.name ilike $2 || $3 || '%'
1198         and bucket_id = $4
1199         and array_length(regexp_split_to_array(objects.name, '/'), 1) = $1
1200     order by ' || v_order_by || '
1201     limit $5
1202     offset $6' using levels, prefix, search, bucketname, limits, offsets;

```

```

1203 end;
1204 $_$;
1205
1206
1207 ALTER FUNCTION storage.search(prefix text, bucketname text, limits integer, levels
integer, offsets integer, search text, sortcolumn text, sortorder text) OWNER TO
supabase_storage_admin;
1208
1209 SET default_tablespace = '';
1210
1211 SET default_table_access_method = heap;
1212
1213 --
1214 -- Name: audit_log_entries; Type: TABLE; Schema: auth; Owner: supabase_auth_admin
1215 --
1216
1217 CREATE TABLE auth.audit_log_entries (
1218     instance_id uuid,
1219     id uuid NOT NULL,
1220     payload json,
1221     created_at timestamp with time zone
1222 );
1223
1224
1225 ALTER TABLE auth.audit_log_entries OWNER TO supabase_auth_admin;
1226
1227 --
1228 -- Name: TABLE audit_log_entries; Type: COMMENT; Schema: auth; Owner: supabase_auth_admin
1229 --
1230
1231 COMMENT ON TABLE auth.audit_log_entries IS 'Auth: Audit trail for user actions.';
1232
1233
1234 --
1235 -- Name: identities; Type: TABLE; Schema: auth; Owner: supabase_auth_admin
1236 --
1237
1238 CREATE TABLE auth.identities (
1239     id text NOT NULL,
1240     user_id uuid NOT NULL,
1241     identity_data jsonb NOT NULL,
1242     provider text NOT NULL,
1243     last_sign_in_at timestamp with time zone,
1244     created_at timestamp with time zone,
1245     updated_at timestamp with time zone
1246 );
1247
1248
1249 ALTER TABLE auth.identities OWNER TO supabase_auth_admin;
1250
1251 --
1252 -- Name: TABLE identities; Type: COMMENT; Schema: auth; Owner: supabase_auth_admin
1253 --
1254
1255 COMMENT ON TABLE auth.identities IS 'Auth: Stores identities associated to a user.';
1256
1257
1258 --
1259 -- Name: instances; Type: TABLE; Schema: auth; Owner: supabase_auth_admin
1260 --
1261
1262 CREATE TABLE auth.instances (
1263     id uuid NOT NULL,
1264     uuid uuid,

```

```
1265     raw_base_config text,
1266     created_at timestamp with time zone,
1267     updated_at timestamp with time zone
1268 );
1269
1270
1271 ALTER TABLE auth.instances OWNER TO supabase_auth_admin;
1272
1273 --
1274 -- Name: TABLE instances; Type: COMMENT; Schema: auth; Owner: supabase_auth_admin
1275 --
1276
1277 COMMENT ON TABLE auth.instances IS 'Auth: Manages users across multiple sites.';
1278
1279
1280 --
1281 -- Name: refresh_tokens; Type: TABLE; Schema: auth; Owner: supabase_auth_admin
1282 --
1283
1284 CREATE TABLE auth.refresh_tokens (
1285     instance_id uuid,
1286     id bigint NOT NULL,
1287     token character varying(255),
1288     user_id character varying(255),
1289     revoked boolean,
1290     created_at timestamp with time zone,
1291     updated_at timestamp with time zone,
1292     parent character varying(255)
1293 );
1294
1295
1296 ALTER TABLE auth.refresh_tokens OWNER TO supabase_auth_admin;
1297
1298 --
1299 -- Name: TABLE refresh_tokens; Type: COMMENT; Schema: auth; Owner: supabase_auth_admin
1300 --
1301
1302 COMMENT ON TABLE auth.refresh_tokens IS 'Auth: Store of tokens used to refresh JWT
tokens once they expire.';
1303
1304
1305 --
1306 -- Name: refresh_tokens_id_seq; Type: SEQUENCE; Schema: auth; Owner: supabase_auth_admin
1307 --
1308
1309 CREATE SEQUENCE auth.refresh_tokens_id_seq
1310     START WITH 1
1311     INCREMENT BY 1
1312     NO MINVALUE
1313     NO MAXVALUE
1314     CACHE 1;
1315
1316
1317 ALTER TABLE auth.refresh_tokens_id_seq OWNER TO supabase_auth_admin;
1318
1319 --
1320 -- Name: refresh_tokens_id_seq; Type: SEQUENCE OWNED BY; Schema: auth; Owner:
supabase_auth_admin
1321 --
1322
1323 ALTER SEQUENCE auth.refresh_tokens_id_seq OWNED BY auth.refresh_tokens.id;
1324
1325
1326 --
```

```

1327 -- Name: schema_migrations; Type: TABLE; Schema: auth; Owner: supabase_auth_admin
1328 --
1329
1330 CREATE TABLE auth.schema_migrations (
1331     version character varying(255) NOT NULL
1332 );
1333
1334
1335 ALTER TABLE auth.schema_migrations OWNER TO supabase_auth_admin;
1336
1337 --
1338 -- Name: TABLE schema_migrations; Type: COMMENT; Schema: auth; Owner: supabase_auth_admin
1339 --
1340
1341 COMMENT ON TABLE auth.schema_migrations IS 'Auth: Manages updates to the auth system.';
1342
1343
1344 --
1345 -- Name: users; Type: TABLE; Schema: auth; Owner: supabase_auth_admin
1346 --
1347
1348 CREATE TABLE auth.users (
1349     instance_id uuid,
1350     id uuid NOT NULL,
1351     aud character varying(255),
1352     role character varying(255),
1353     email character varying(255),
1354     encrypted_password character varying(255),
1355     email_confirmed_at timestamp with time zone,
1356     invited_at timestamp with time zone,
1357     confirmation_token character varying(255),
1358     confirmation_sent_at timestamp with time zone,
1359     recovery_token character varying(255),
1360     recovery_sent_at timestamp with time zone,
1361     email_change_token_new character varying(255),
1362     email_change character varying(255),
1363     email_change_sent_at timestamp with time zone,
1364     last_sign_in_at timestamp with time zone,
1365     raw_app_meta_data jsonb,
1366     raw_user_meta_data jsonb,
1367     is_super_admin boolean,
1368     created_at timestamp with time zone,
1369     updated_at timestamp with time zone,
1370     phone character varying(15) DEFAULT NULL::character varying,
1371     phone_confirmed_at timestamp with time zone,
1372     phone_change character varying(15) DEFAULT ''::character varying,
1373     phone_change_token character varying(255) DEFAULT ''::character varying,
1374     phone_change_sent_at timestamp with time zone,
1375     confirmed_at timestamp with time zone GENERATED ALWAYS AS (LEAST(email_confirmed_at,
1376         phone_confirmed_at)) STORED,
1377     email_change_token_current character varying(255) DEFAULT ''::character varying,
1378     email_change_confirm_status smallint DEFAULT 0,
1379     banned_until timestamp with time zone,
1380     reauthentication_token character varying(255) DEFAULT ''::character varying,
1381     reauthentication_sent_at timestamp with time zone,
1382     CONSTRAINT users_email_change_confirm_status_check CHECK (((
1383         email_change_confirm_status >= 0) AND (email_change_confirm_status <= 2)))
1384 );
1385
1386
1387 ALTER TABLE auth.users OWNER TO supabase_auth_admin;
1388
1389 --
1390 -- Name: TABLE users; Type: COMMENT; Schema: auth; Owner: supabase_auth_admin

```

```

1389  --
1390
1391  COMMENT ON TABLE auth.users IS 'Auth: Stores user login data within a secure schema.';
1392
1393
1394  --
1395  -- Name: Decisions; Type: TABLE; Schema: public; Owner: supabase_admin
1396  --
1397
1398  CREATE TABLE public."Decisions" (
1399      id bigint NOT NULL,
1400      project_id bigint,
1401      voting_result double precision,
1402      is_project_accepted boolean,
1403      date_time_timezone timestamp with time zone
1404  );
1405
1406
1407  ALTER TABLE public."Decisions" OWNER TO supabase_admin;
1408
1409  --
1410  -- Name: COLUMN "Decisions".voting_result; Type: COMMENT; Schema: public; Owner:
1411  supabase_admin
1412  --
1413
1414  COMMENT ON COLUMN public."Decisions".voting_result IS 'Percentage (out of 100) of
1415  accept votes among all the SingleVotes for that project (ex: 80)';
1416
1417  --
1418  -- Name: COLUMN "Decisions".is_project_accepted; Type: COMMENT; Schema: public; Owner:
1419  supabase_admin
1420  --
1421
1422  COMMENT ON COLUMN public."Decisions".is_project_accepted IS 'Accept or Reject decision.
1423  if true, if means the project proposal is accepted.';
1424
1425  --
1426  -- Name: COLUMN "Decisions".date_time_timezone; Type: COMMENT; Schema: public; Owner:
1427  supabase_admin
1428  --
1429
1430  COMMENT ON COLUMN public."Decisions".date_time_timezone IS 'Date, time, and timezone of
1431  when the voting decision has been reached.';
1432
1433  --
1434  -- Name: Decision_id_seq; Type: SEQUENCE; Schema: public; Owner: supabase_admin
1435  --
1436
1437  ALTER TABLE public."Decisions" ALTER COLUMN id ADD GENERATED BY DEFAULT AS IDENTITY (
1438      SEQUENCE NAME public."Decision_id_seq"
1439      START WITH 1
1440      INCREMENT BY 1
1441      NO MINVALUE
1442      NO MAXVALUE
1443      CACHE 1
1444  );
1445
1446  --
1447  -- Name: Donations; Type: TABLE; Schema: public; Owner: supabase_admin
1448  --

```



```

1447
1448 CREATE TABLE public."Donations" (
1449     id bigint NOT NULL,
1450     donor_id bigint,
1451     project_id bigint,
1452     proof_of_completion_file character varying,
1453     project_completion_date_time_timezone timestamp with time zone
1454 );
1455
1456
1457 ALTER TABLE public."Donations" OWNER TO supabase_admin;
1458
1459 --
1460 -- Name: COLUMN "Donations".proof_of_completion_file; Type: COMMENT; Schema: public;
1461 -- Owner: supabase_admin
1462 --
1463 COMMENT ON COLUMN public."Donations".proof_of_completion_file IS 'The name of the file
1464 which shows proof that the project has been completed.';
1465
1466 --
1467 -- Name: COLUMN "Donations".project_completion_date_time_timezone; Type: COMMENT;
1468 -- Schema: public; Owner: supabase_admin
1469 --
1470 COMMENT ON COLUMN public."Donations".project_completion_date_time_timezone IS 'Date,
1471 time, and timezone of when the project was completed, as taken from the time of
1472 submission of the proof of completion document.';
1473
1474 --
1475 -- Name: Donation_id_seq; Type: SEQUENCE; Schema: public; Owner: supabase_admin
1476 --
1477 ALTER TABLE public."Donations" ALTER COLUMN id ADD GENERATED BY DEFAULT AS IDENTITY (
1478     SEQUENCE NAME public."Donation_id_seq"
1479     START WITH 1
1480     INCREMENT BY 1
1481     NO MINVALUE
1482     NO MAXVALUE
1483     CACHE 1
1484 );
1485
1486
1487 --
1488 -- Name: Donors; Type: TABLE; Schema: public; Owner: supabase_admin
1489 --
1490
1491 CREATE TABLE public."Donors" (
1492     id bigint NOT NULL,
1493     wallet_address character varying
1494 );
1495
1496
1497 ALTER TABLE public."Donors" OWNER TO supabase_admin;
1498
1499 --
1500 -- Name: Donor_id_seq; Type: SEQUENCE; Schema: public; Owner: supabase_admin
1501 --
1502
1503 ALTER TABLE public."Donors" ALTER COLUMN id ADD GENERATED BY DEFAULT AS IDENTITY (
1504     SEQUENCE NAME public."Donor_id_seq"
1505     START WITH 1

```

```

1506         INCREMENT BY 1
1507     NO MINVALUE
1508     NO MAXVALUE
1509     CACHE 1
1510 );
1511
1512
1513 --
1514 -- Name: Governors; Type: TABLE; Schema: public; Owner: supabase_admin
1515 --
1516
1517 CREATE TABLE public."Governors" (
1518     id bigint NOT NULL,
1519     wallet_address character varying
1520 );
1521
1522
1523 ALTER TABLE public."Governors" OWNER TO supabase_admin;
1524
1525 --
1526 -- Name: Governors_id_seq; Type: SEQUENCE; Schema: public; Owner: supabase_admin
1527 --
1528
1529 ALTER TABLE public."Governors" ALTER COLUMN id ADD GENERATED BY DEFAULT AS IDENTITY (
1530     SEQUENCE NAME public."Governors_id_seq"
1531     START WITH 1
1532     INCREMENT BY 1
1533     NO MINVALUE
1534     NO MAXVALUE
1535     CACHE 1
1536 );
1537
1538
1539 --
1540 -- Name: Organizations; Type: TABLE; Schema: public; Owner: supabase_admin
1541 --
1542
1543 CREATE TABLE public."Organizations" (
1544     id bigint NOT NULL,
1545     name character varying,
1546     website character varying,
1547     country character varying,
1548     active boolean,
1549     other_info text,
1550     image_file character varying,
1551     wallet_address character varying
1552 );
1553
1554
1555 ALTER TABLE public."Organizations" OWNER TO supabase_admin;
1556
1557 --
1558 -- Name: TABLE "Organizations"; Type: COMMENT; Schema: public; Owner: supabase_admin
1559 --
1560
1561 COMMENT ON TABLE public."Organizations" IS 'Charity Organizations';
1562
1563
1564 --
1565 -- Name: COLUMN "Organizations".other_info; Type: COMMENT; Schema: public; Owner:
supabase_admin
1566 --
1567
1568 COMMENT ON COLUMN public."Organizations".other_info IS 'Name of the file which contains

```

```

more information about the Organization';
1569
1570
1571 --
1572 -- Name: COLUMN "Organizations".image_file; Type: COMMENT; Schema: public; Owner:
supabase_admin
1573 --
1574
1575 COMMENT ON COLUMN public."Organizations".image_file IS 'Name of the image file for the
organization. 1080x1080 resolution/size';
1576
1577
1578 --
1579 -- Name: COLUMN "Organizations".wallet_address; Type: COMMENT; Schema: public; Owner:
supabase_admin
1580 --
1581
1582 COMMENT ON COLUMN public."Organizations".wallet_address IS 'Wallet address for the
organization, to which the funding for the project will be deposited.';
1583
1584
1585 --
1586 -- Name: Organizations_id_seq; Type: SEQUENCE; Schema: public; Owner: supabase_admin
1587 --
1588
1589 ALTER TABLE public."Organizations" ALTER COLUMN id ADD GENERATED BY DEFAULT AS IDENTITY (
1590     SEQUENCE NAME public."Organizations_id_seq"
1591     START WITH 1
1592     INCREMENT BY 1
1593     NO MINVALUE
1594     NO MAXVALUE
1595     CACHE 1
1596 );
1597
1598
1599 --
1600 -- Name: Projects; Type: TABLE; Schema: public; Owner: supabase_admin
1601 --
1602
1603 CREATE TABLE public."Projects" (
1604     id bigint NOT NULL,
1605     solution_id bigint,
1606     organization_id bigint,
1607     budget_usd double precision,
1608     other_info text,
1609     country character varying,
1610     project_duration_days bigint,
1611     status character varying,
1612     "date_time_timezone" timestamp with time zone,
1613     "mintPriceHBAR" double precision,
1614     "maxNFTSupply" bigint
1615 );
1616
1617
1618 ALTER TABLE public."Projects" OWNER TO supabase_admin;
1619
1620 --
1621 -- Name: TABLE "Projects"; Type: COMMENT; Schema: public; Owner: supabase_admin
1622 --
1623
1624 COMMENT ON TABLE public."Projects" IS 'Projects created by organizations implementing
solutions';
1625
1626

```

```
1627  --
1628  -- Name: COLUMN "Projects".country; Type: COMMENT; Schema: public; Owner: supabase_admin
1629  --
1630
1631  COMMENT ON COLUMN public."Projects".country IS 'Country where the project is to be
applied';
1632
1633
1634  --
1635  -- Name: COLUMN "Projects".project_duration_days; Type: COMMENT; Schema: public; Owner:
supabase_admin
1636  --
1637
1638  COMMENT ON COLUMN public."Projects".project_duration_days IS 'Number of days that it
will take for the project to be completed';
1639
1640
1641  --
1642  -- Name: COLUMN "Projects".status; Type: COMMENT; Schema: public; Owner: supabase_admin
1643  --
1644
1645  COMMENT ON COLUMN public."Projects".status IS 'Possible Values: UnderReview, Rejected,
Accepted, Initiated, Uninitiated, Completed, Incomplete';
1646
1647
1648  --
1649  -- Name: COLUMN "Projects"."date_time_timezone "; Type: COMMENT; Schema: public; Owner:
supabase_admin
1650  --
1651
1652  COMMENT ON COLUMN public."Projects"."date_time_timezone " IS 'Date, time and timezone
of when the project was submitted for voting';
1653
1654
1655  --
1656  -- Name: COLUMN "Projects"."mintPriceHBAR"; Type: COMMENT; Schema: public; Owner:
supabase_admin
1657  --
1658
1659  COMMENT ON COLUMN public."Projects"."mintPriceHBAR" IS 'donation amount in HBAR to mint
a single donation NFT';
1660
1661
1662  --
1663  -- Name: COLUMN "Projects"."maxNFTSupply"; Type: COMMENT; Schema: public; Owner:
supabase_admin
1664  --
1665
1666  COMMENT ON COLUMN public."Projects"."maxNFTSupply" IS 'max number of donation NFTs for
completion the full donation for the solution';
1667
1668
1669  --
1670  -- Name: Projects_id_seq; Type: SEQUENCE; Schema: public; Owner: supabase_admin
1671  --
1672
1673  ALTER TABLE public."Projects" ALTER COLUMN id ADD GENERATED BY DEFAULT AS IDENTITY (
1674      SEQUENCE NAME public."Projects_id_seq"
1675      START WITH 1
1676      INCREMENT BY 1
1677      NO MINVALUE
1678      NO MAXVALUE
1679      CACHE 1
1680  );
```

```
1681
1682
1683 --
1684 -- Name: SingleVotes; Type: TABLE; Schema: public; Owner: supabase_admin
1685 --
1686
1687 CREATE TABLE public."SingleVotes" (
1688     id bigint NOT NULL,
1689     governor_id bigint,
1690     vote_value bigint,
1691     project_id bigint,
1692     date_time_timezone timestamp with time zone
1693 );
1694
1695
1696 ALTER TABLE public."SingleVotes" OWNER TO supabase_admin;
1697
1698 --
1699 -- Name: COLUMN "SingleVotes".governor_id; Type: COMMENT; Schema: public; Owner:
supabase_admin
1700 --
1701
1702 COMMENT ON COLUMN public."SingleVotes".governor_id IS 'id of the Governor who is
casting the vote';
1703
1704
1705 --
1706 -- Name: COLUMN "SingleVotes".vote_value; Type: COMMENT; Schema: public; Owner:
supabase_admin
1707 --
1708
1709 COMMENT ON COLUMN public."SingleVotes".vote_value IS 'Value of the vote. Initially
either 0 or 1, where 0 means reject and 1 means accept';
1710
1711
1712 --
1713 -- Name: COLUMN "SingleVotes".project_id; Type: COMMENT; Schema: public; Owner:
supabase_admin
1714 --
1715
1716 COMMENT ON COLUMN public."SingleVotes".project_id IS 'Project which is under vote';
1717
1718
1719 --
1720 -- Name: COLUMN "SingleVotes".date_time_timezone; Type: COMMENT; Schema: public; Owner:
supabase_admin
1721 --
1722
1723 COMMENT ON COLUMN public."SingleVotes".date_time_timezone IS 'Date, time, and timezone
of when the vote was casted';
1724
1725
1726 --
1727 -- Name: SingleVote_id_seq; Type: SEQUENCE; Schema: public; Owner: supabase_admin
1728 --
1729
1730 ALTER TABLE public."SingleVotes" ALTER COLUMN id ADD GENERATED BY DEFAULT AS IDENTITY (
1731     SEQUENCE NAME public."SingleVote_id_seq"
1732     START WITH 1
1733     INCREMENT BY 1
1734     NO MINVALUE
1735     NO MAXVALUE
1736     CACHE 1
1737 );
```

```

1738
1739
1740 --
1741 -- Name: Solutions; Type: TABLE; Schema: public; Owner: supabase_admin
1742 --
1743
1744 CREATE TABLE public."Solutions" (
1745     id bigint NOT NULL,
1746     name character varying,
1747     category character varying,
1748     nature character varying,
1749     summary character varying,
1750     estimated_cost_usd double precision,
1751     cost_breakdown character varying,
1752     information_source character varying,
1753     application_scenarios character varying,
1754     applicable_in_city boolean,
1755     applicable_in_nature boolean,
1756     low_tech bigint,
1757     simplicity bigint,
1758     low_cost bigint,
1759     portability bigint,
1760     versatility bigint,
1761     local_materials_required character varying,
1762     labour_hours_required bigint,
1763     product_company_name character varying,
1764     product_webpage character varying,
1765     other_info text,
1766     applied_by_organizations character varying,
1767     image_file_name character varying
1768 );
1769
1770
1771 ALTER TABLE public."Solutions" OWNER TO supabase_admin;
1772
1773 --
1774 -- Name: TABLE "Solutions"; Type: COMMENT; Schema: public; Owner: supabase_admin
1775 --
1776
1777 COMMENT ON TABLE public."Solutions" IS 'Possible solutions that can be used to help
people';
1778
1779
1780 --
1781 -- Name: COLUMN "Solutions".image_file_name; Type: COMMENT; Schema: public; Owner:
supabase_admin
1782 --
1783
1784 COMMENT ON COLUMN public."Solutions".image_file_name IS 'Name of the image file for the
Solution';
1785
1786
1787 --
1788 -- Name: Solutions_id_seq; Type: SEQUENCE; Schema: public; Owner: supabase_admin
1789 --
1790
1791 ALTER TABLE public."Solutions" ALTER COLUMN id ADD GENERATED BY DEFAULT AS IDENTITY (
1792     SEQUENCE NAME public."Solutions_id_seq"
1793     START WITH 1
1794     INCREMENT BY 1
1795     NO MINVALUE
1796     NO MAXVALUE
1797     CACHE 1
1798 );

```

```

1799
1800
1801 --
1802 -- Name: SystemConstants; Type: TABLE; Schema: public; Owner: supabase_admin
1803 --
1804
1805 CREATE TABLE public."SystemConstants" (
1806     id bigint NOT NULL,
1807     min_perc_vote double precision,
1808     path_reward_per_vote bigint,
1809     hours_to_vote bigint,
1810     days_to_claim_donation bigint,
1811     days_to_complete_project bigint
1812 );
1813
1814
1815 ALTER TABLE public."SystemConstants" OWNER TO supabase_admin;
1816
1817 --
1818 -- Name: TABLE "SystemConstants"; Type: COMMENT; Schema: public; Owner: supabase_admin
1819 --
1820
1821 COMMENT ON TABLE public."SystemConstants" IS 'Constants of the system';
1822
1823
1824 --
1825 -- Name: COLUMN "SystemConstants".min_perc_vote; Type: COMMENT; Schema: public; Owner:
supabase_admin
1826 --
1827
1828 COMMENT ON COLUMN public."SystemConstants".min_perc_vote IS 'between 50 and 100, this
is the percentage of votes required by a proposal to pass';
1829
1830
1831 --
1832 -- Name: COLUMN "SystemConstants".path_reward_per_vote; Type: COMMENT; Schema: public;
Owner: supabase_admin
1833 --
1834
1835 COMMENT ON COLUMN public."SystemConstants".path_reward_per_vote IS 'number of PATH
tokens distributed to each governor each time they submit a vote';
1836
1837
1838 --
1839 -- Name: COLUMN "SystemConstants".hours_to_vote; Type: COMMENT; Schema: public; Owner:
supabase_admin
1840 --
1841
1842 COMMENT ON COLUMN public."SystemConstants".hours_to_vote IS 'number of hours given to
Governors to submit a vote for a proposal, once they are invited to vote';
1843
1844
1845 --
1846 -- Name: COLUMN "SystemConstants".days_to_claim_donation; Type: COMMENT; Schema:
public; Owner: supabase_admin
1847 --
1848
1849 COMMENT ON COLUMN public."SystemConstants".days_to_claim_donation IS 'number of days
given to an Organization to claim the donation and start the project';
1850
1851
1852 --
1853 -- Name: COLUMN "SystemConstants".days_to_complete_project; Type: COMMENT; Schema:
public; Owner: supabase_admin

```

```

1854  --
1855
1856  COMMENT ON COLUMN public."SystemConstants".days_to_complete_project IS 'number of days
given to an Organization to complete the project and submit the proof of completion
report for a project, after they claimed the donation';
1857
1858
1859  --
1860  -- Name: SystemConstants_id_seq; Type: SEQUENCE; Schema: public; Owner: supabase_admin
1861  --
1862
1863  ALTER TABLE public."SystemConstants" ALTER COLUMN id ADD GENERATED BY DEFAULT AS
IDENTITY (
1864      SEQUENCE NAME public."SystemConstants_id_seq"
1865      START WITH 1
1866      INCREMENT BY 1
1867      NO MINVALUE
1868      NO MAXVALUE
1869      CACHE 1
1870  );
1871
1872
1873  --
1874  -- Name: schema_migrations; Type: TABLE; Schema: realtime; Owner: supabase_admin
1875  --
1876
1877  CREATE TABLE realtime.schema_migrations (
1878      version bigint NOT NULL,
1879      inserted_at timestamp(0) without time zone
1880  );
1881
1882
1883  ALTER TABLE realtime.schema_migrations OWNER TO supabase_admin;
1884
1885  --
1886  -- Name: subscription; Type: TABLE; Schema: realtime; Owner: supabase_admin
1887  --
1888
1889  CREATE TABLE realtime.subscription (
1890      id bigint NOT NULL,
1891      subscription_id uuid NOT NULL,
1892      entity regclass NOT NULL,
1893      filters realtime.user_defined_filter[] DEFAULT '{}'::realtime.user_defined_filter[]
NOT NULL,
1894      claims jsonb NOT NULL,
1895      claims_role regrole GENERATED ALWAYS AS (realtime.to_regrole((claims ->> 'role'::
text))) STORED NOT NULL,
1896      created_at timestamp without time zone DEFAULT timezone('utc'::text, now()) NOT NULL
1897  );
1898
1899
1900  ALTER TABLE realtime.subscription OWNER TO supabase_admin;
1901
1902  --
1903  -- Name: subscription_id_seq; Type: SEQUENCE; Schema: realtime; Owner: supabase_admin
1904  --
1905
1906  ALTER TABLE realtime.subscription ALTER COLUMN id ADD GENERATED ALWAYS AS IDENTITY (
1907      SEQUENCE NAME realtime.subscription_id_seq
1908      START WITH 1
1909      INCREMENT BY 1
1910      NO MINVALUE
1911      NO MAXVALUE
1912      CACHE 1

```



```

1913 );
1914
1915
1916 --
1917 -- Name: buckets; Type: TABLE; Schema: storage; Owner: supabase_storage_admin
1918 --
1919
1920 CREATE TABLE storage.buckets (
1921     id text NOT NULL,
1922     name text NOT NULL,
1923     owner uuid,
1924     created_at timestamp with time zone DEFAULT now(),
1925     updated_at timestamp with time zone DEFAULT now(),
1926     public boolean DEFAULT false
1927 );
1928
1929
1930 ALTER TABLE storage.buckets OWNER TO supabase_storage_admin;
1931
1932 --
1933 -- Name: migrations; Type: TABLE; Schema: storage; Owner: supabase_storage_admin
1934 --
1935
1936 CREATE TABLE storage.migrations (
1937     id integer NOT NULL,
1938     name character varying(100) NOT NULL,
1939     hash character varying(40) NOT NULL,
1940     executed_at timestamp without time zone DEFAULT CURRENT_TIMESTAMP
1941 );
1942
1943
1944 ALTER TABLE storage.migrations OWNER TO supabase_storage_admin;
1945
1946 --
1947 -- Name: objects; Type: TABLE; Schema: storage; Owner: supabase_storage_admin
1948 --
1949
1950 CREATE TABLE storage.objects (
1951     id uuid DEFAULT extensions.uuid_generate_v4() NOT NULL,
1952     bucket_id text,
1953     name text,
1954     owner uuid,
1955     created_at timestamp with time zone DEFAULT now(),
1956     updated_at timestamp with time zone DEFAULT now(),
1957     last_accessed_at timestamp with time zone DEFAULT now(),
1958     metadata jsonb,
1959     path_tokens text[] GENERATED ALWAYS AS (string_to_array(name, '/'::text)) STORED
1960 );
1961
1962
1963 ALTER TABLE storage.objects OWNER TO supabase_storage_admin;
1964
1965 --
1966 -- Name: refresh_tokens id; Type: DEFAULT; Schema: auth; Owner: supabase_auth_admin
1967 --
1968
1969 ALTER TABLE ONLY auth.refresh_tokens ALTER COLUMN id SET DEFAULT nextval(
    'auth.refresh_tokens_id_seq'::regclass);
1970
1971
1972 --
1973 -- Data for Name: audit_log_entries; Type: TABLE DATA; Schema: auth; Owner:
    supabase_auth_admin
1974 --

```

```
1975
1976 COPY auth.audit_log_entries (instance_id, id, payload, created_at) FROM stdin;
1977 \.
1978
1979
1980 --
1981 -- Data for Name: identities; Type: TABLE DATA; Schema: auth; Owner: supabase_auth_admin
1982 --
1983
1984 COPY auth.identities (id, user_id, identity_data, provider, last_sign_in_at, created_at,
1985   updated_at) FROM stdin;
1986 \.
1987
1988 --
1989 -- Data for Name: instances; Type: TABLE DATA; Schema: auth; Owner: supabase_auth_admin
1990 --
1991
1992 COPY auth.instances (id, uuid, raw_base_config, created_at, updated_at) FROM stdin;
1993 \.
1994
1995
1996 --
1997 -- Data for Name: refresh_tokens; Type: TABLE DATA; Schema: auth; Owner:
1998   supabase_auth_admin
1999 --
2000
2001 COPY auth.refresh_tokens (instance_id, id, token, user_id, revoked, created_at,
2002   updated_at, parent) FROM stdin;
2003 \.
2004
2005
2006 --
2007 -- Data for Name: schema_migrations; Type: TABLE DATA; Schema: auth; Owner:
2008   supabase_auth_admin
2009 --
2010
2011 COPY auth.schema_migrations (version) FROM stdin;
2012 20171026211738
2013 20171026211808
2014 20171026211834
2015 20180103212743
2016 20180108183307
2017 20180119214651
2018 20180125194653
2019 00
2020 20210710035447
2021 20210722035447
2022 20210730183235
2023 20210909172000
2024 20210927181326
2025 20211122151130
2026 20211124214934
2027 20211202183645
2028 20220114185221
2029 20220114185340
2030 20220224000811
2031 20220323170000
2032 20220429102000
2033 \.
2034
2035
2036 --
2037 -- Data for Name: users; Type: TABLE DATA; Schema: auth; Owner: supabase_auth_admin
```

```

2035  --
2036
2037 COPY auth.users (instance_id, id, aud, role, email, encrypted_password,
email_confirmed_at, invited_at, confirmation_token, confirmation_sent_at, recovery_token
, recovery_sent_at, email_change_token_new, email_change, email_change_sent_at,
last_sign_in_at, raw_app_meta_data, raw_user_meta_data, is_super_admin, created_at,
updated_at, phone, phone_confirmed_at, phone_change, phone_change_token,
phone_change_sent_at, email_change_token_current, email_change_confirm_status,
banned_until, reauthentication_token, reauthentication_sent_at) FROM stdin;
2038 \.
2039
2040
2041  --
2042  -- Data for Name: Decisions; Type: TABLE DATA; Schema: public; Owner: supabase_admin
2043  --
2044
2045 COPY public."Decisions" (id, project_id, voting_result, is_project_accepted,
date_time_timezone) FROM stdin;
2046 4    4    66.67    t    2022-04-27 08:11:43+00
2047 1    2    66.67    t    2022-05-05 09:12:01+00
2048 2    3    100 t    2022-04-24 19:09:45+00
2049 3    5    100 t    2022-04-24 19:12:23+00
2050 5    9    100 t    2022-05-12 11:42:13+00
2051 6    6    100 t    2022-05-11 11:42:51+00
2052 7    7    100 t    2022-05-11 02:47:36+00
2053 8    8    100 t    2022-05-16 11:49:17+00
2054 9    1    0    f    2022-05-06 05:50:09+00
2055 \.
2056
2057
2058  --
2059  -- Data for Name: Donations; Type: TABLE DATA; Schema: public; Owner: supabase_admin
2060  --
2061
2062 COPY public."Donations" (id, donor_id, project_id, proof_of_completion_file,
project_completion_date_time_timezone) FROM stdin;
2063 1    1    5    \N    \N
2064 \.
2065
2066
2067  --
2068  -- Data for Name: Donors; Type: TABLE DATA; Schema: public; Owner: supabase_admin
2069  --
2070
2071 COPY public."Donors" (id, wallet_address) FROM stdin;
2072 1    0.0.34735190
2073 \.
2074
2075
2076  --
2077  -- Data for Name: Governors; Type: TABLE DATA; Schema: public; Owner: supabase_admin
2078  --
2079
2080 COPY public."Governors" (id, wallet_address) FROM stdin;
2081 1    0.0.34735139
2082 2    0.0.34735159
2083 3    0.0.34735165
2084 \.
2085
2086
2087  --
2088  -- Data for Name: Organizations; Type: TABLE DATA; Schema: public; Owner: supabase_admin
2089  --
2090

```

```

2091 COPY public."Organizations" (id, name, website, country, active, other_info, image_file,
2092     wallet_address) FROM stdin;
2093 1   Water From Air Foundation   https://waterfromair.org.tr Turkey   t   Lack of access
2094     to water is the root of many economic and social problems around the world. Our charity
2095     tackles and aims to resolve this problems for communities around the world. https://
2096     epvzhrgranrzwexmiksg.supabase.co/storage/v1/object/public/images/R001.jpg 0.0.34735175
2097 2   Boon Lay Goodness Movement https://boonlaygoodness.com Singapore   t   Our charity
2098     organization is to reduce all types of suffering for people around the world. Dont eye
2099     power. Instead, join us and contribute. https://epvzhrgranrzwexmiksg.supabase.co/
2100     storage/v1/object/public/images/R002.jpg 0.0.34735181
2101 3   Lagos United Charity https://lagosunitedcharity.org Nigeria t   Lagos United
2102     Charity aims to alleviate poverty in Lagos and its suburbs. Rather than coordinating
2103     direct food donations, our goal is to provide the tools to reduce poverty. https://
2104     epvzhrgranrzwexmiksg.supabase.co/storage/v1/object/public/images/R003.jpg 0.0.34735186
2105 \.
2106
2107 --
2108 -- Data for Name: Projects; Type: TABLE DATA; Schema: public; Owner: supabase_admin
2109 --
2110
2111 COPY public."Projects" (id, solution_id, organization_id, budget_usd, other_info,
2112     country, project_duration_days, status, "date_time_timezone ", "mintPriceHBAR",
2113     "maxNFTSupply") FROM stdin;
2114
2115 1   1   1   2500   \N   Indonesia   35   Accepted   2022-05-03 18:20:23+00   1000   27
2116 3   1   1   3000   \N   Peru   30   Accepted   2022-05-02 02:24:42+00   2000   16
2117 4   2   2   3000   \N   Bangladesh   45   Accepted   2022-04-26 17:25:03+00   2000   16
2118 6   1   1   3500   \N   India   60   Accepted   2022-05-10 08:22:05+00   2000   19
2119 7   2   2   3200   \N   Malaysia   50   Accepted   2022-05-10 10:20:18+00   5000   7
2120 8   3   2   1600   \N   Singapore   23   Accepted   2022-05-13 15:00:17+00   1000   18
2121 5   7   3   2800   \N   Nigeria   55   Initiated   2022-04-22 16:24:00+00   2000   15
2122 2   4   2   2500   \N   Vietnam   40   Accepted   2022-05-08 16:21:24+00   1000   27
2123 9   9   3   2000   \N   India   20   Accepted   2022-05-11 19:33:54+00   2000   11
2124 \.
2125
2126 --
2127 -- Data for Name: SingleVotes; Type: TABLE DATA; Schema: public; Owner: supabase_admin
2128 --
2129
2130 COPY public."SingleVotes" (id, governor_id, vote_value, project_id, date_time_timezone)
2131 FROM stdin;
2132 1   1   0   1   2022-05-07 19:00:50+00
2133 5   3   1   2   2022-05-01 12:04:35+00
2134 6   2   1   3   2022-05-02 08:01:46+00
2135 7   3   1   3   2022-05-02 08:03:33+00
2136 8   1   1   4   2022-04-24 11:06:37+00
2137 3   1   1   2   2022-05-01 08:59:16+00
2138 9   2   1   4   2022-04-24 10:08:20+00
2139 4   2   0   2   2022-05-01 19:00:58+00
2140 10  3   0   4   2022-04-24 19:03:41+00
2141 11  1   1   5   2022-04-21 19:05:37+00
2142 12  2   1   5   2022-04-20 19:06:07+00
2143 2   2   0   1   2022-05-04 08:00:20+00
2144 13  3   1   9   2022-05-12 19:35:49+00
2145 14  2   1   6   2022-05-10 05:42:00+00
2146 15  1   1   7   2022-05-10 12:43:30+00
2147 16  3   1   7   2022-05-10 16:44:12+00
2148 17  1   1   8   2022-05-14 02:48:47+00
2149 \.
2150
2151 --
2152 -- Data for Name: Solutions; Type: TABLE DATA; Schema: public; Owner: supabase_admin

```

2142 --

2143

2144 COPY public."Solutions" (id, name, category, nature, summary, estimated_cost_usd, cost_breakdown, information_source, application_scenarios, applicable_in_city, applicable_in_nature, low_tech, simplicity, low_cost, portability, versatility, local_materials_required, labour_hours_required, product_company_name, product_webpage, other_info, applied_by_organizations, image_file_name) FROM stdin;

2145 7 ecoDrone Soil Generating UAV (Unmanned Aerial Vehicle) that can drop 2500 seedbombs in 10 minutes 3200 2200 USD for the drone, 1000 USD for seed bombs <https://youtu.be/byFrZ1T3EHI> Empty land with reasonable rainfall, which still has soil with enough quality to grow plants and trees f t 5 6 6 7 7 \N 5 Ecording <https://ecording.org/en/> <https://ecording.org/en/faq/> Ecording <https://epvzhrganrzwfxmiksg.supabase.co/storage/v1/object/public/images/S007.jpg?t=2022-05-14T17:40:19.655Z>

2146 2 HyPump Water Transporting Hydro-powered pump that can pump water to farmers over a mile away, without requiring any electricity for operation 1500 Estimated value of one pump from the manufacturer is 1500 USD, including worldwide shipment. <https://youtu.be/lPDvjIk5cQY> Built on rivers and streams in areas where there is no access to electricity. f t 6 7 8 7 7 Pickaxe, shovel, wheelbarrow 50 \N \N <https://scalingfrontierinnovation.org/spotlights/aqysta-barsha-pump/> aQysta <https://epvzhrganrzwfxmiksg.supabase.co/storage/v1/object/public/images/S002.jpg>

2147 5 WaterLily Energy Generating Generating energy for USB compatible devices 400 Turbine is 190 USD per piece, plus 10 USD for cable, 100 USD for wind kit, 100 USD for tripod <https://youtu.be/rxeMwlpagRI> Windy locations or flowing waters, best for people staying and moving in nature t t 7 8 8 9 6 \N 1 WaterLily <https://www.waterlilyturbine.com/> \N \N <https://epvzhrganrzwfxmiksg.supabase.co/storage/v1/object/public/images/S005.jpg?t=2022-05-14T17:40:02.621Z>

2148 1 Fog catchers Water Generating Water generation using nets in misty locations 2300 100 USD for each net, 15 nets minimum, giving 1500 USD for nets. Then another 750 USD for water tank and piping. 2250 USD total. <https://youtu.be/YxRONAZoMDk> Higher elevated remote locations with myst, or very mysty locations f t 9 8 7 7 7 Wooden poles and nets, hammer and nails, hand tools for piping 100 \N \N \N \N <https://epvzhrganrzwfxmiksg.supabase.co/storage/v1/object/public/images/S001.jpg\r\n>

2149 3 Water-Gate Water Controlling Inflatable portable mini-dam that stops floods and other excess flow of water 2000 2000 USD for each Water-Gate portable dam <https://youtu.be/arZbmV9vqug> Positioned on rivers, water currents, lake sides, and dams, when there is excess water flow t t 8 7 7 7 8 \N 3 MegaSecur <https://water-gate.com/en/barriers/see-all-barriers/> <https://megasecur.com/en/services/advisory-team/> \N <https://epvzhrganrzwfxmiksg.supabase.co/storage/v1/object/public/images/S003.jpg?t=2022-05-14T12:56:53.987Z>

2150 4 Warka Water Water Generating Bamboo tower, with mesh net inside, to cultivate water from fog or rain water 1500 1450 USD for the mesh net that collects the water, 50 USD for ropes <https://youtu.be/U3npQeku4zA> Humid foggy remote locations that do not have reliable water source but humidity in the air f t 8 8 8 5 6 Local Bamboo 30 Warka Water <https://www.warkawater.org/\n> \N Warka Water <https://epvzhrganrzwfxmiksg.supabase.co/storage/v1/object/public/images/S004.jpg?t=2022-05-14T13:10:10.091Z>

2151 6 Solar Cooker Energy Generating Cooker that harvests solar energy into heat 500 500 USD for each cooker <https://youtu.be/7vvCF3TZ9GE> Locations exposed to sunlight t t 9 9 9 7 9 Screwdriver, fastener 3 Solar Chef <https://solar-chef.com/> <https://www.solarcookers.org/resources/download> Solar Cookers International <https://epvzhrganrzwfxmiksg.supabase.co/storage/v1/object/public/images/S006.jpg?t=2022-05-14T17:40:15.330Z>

2152 8 Water-Saving Nozzle Water Saving Sink faucet nozzle that saves up to 98 percent water 1000 50 USD per nozzle, set of 20 nozzles for a building or facility <https://youtu.be/AWRxhbKWvZk> Reliable water piping system with minimum water flow pressure of 3 BAR t f 9 9 9 10 9 \N 1 Altered <https://alteredcompany.com/collections/nozzles> <https://alteredcompany.com/collections/nozzles> Altered <https://epvzhrganrzwfxmiksg.supabase.co/storage/v1/object/public/images/S008.jpg?t=2022-05-14T19:27:27.905Z>

2153 9 Clay Refrigerator Food Storing Zero-electricity clay refrigerator for preserving food and medicine 1500 150 USD for each clay refrigerator, set of 10 refrigerators for poor neighborhoods <https://youtu.be/WPYzV64dUuU> Remote areas with no electricity or poor neighborhoods, in relatively hot regions t t 9 9 8

```

6      8      \N      2      Mitti Cool      https://mitticool.com/product/mitticool-clay-refrigerator50-
liter/      https://mitticool.com/product/mitticool-clay-refrigerator50-liter/      Mitti Cool
https://epvzhrgranrzfwexmiksg.supabase.co/storage/v1/object/public/images/S009.jpg?t=2022
-05-14T19:27:32.635Z
2154      \.
2155
2156
2157      --
2158      -- Data for Name: SystemConstants; Type: TABLE DATA; Schema: public; Owner:
supabase_admin
2159      --
2160
2161      COPY public."SystemConstants" (id, min_perc_vote, path_reward_per_vote, hours_to_vote,
days_to_claim_donation, days_to_complete_project) FROM stdin;
2162      1      70      1      120      7      120
2163      \.
2164
2165
2166      --
2167      -- Data for Name: schema_migrations; Type: TABLE DATA; Schema: realtime; Owner:
supabase_admin
2168      --
2169
2170      COPY realtime.schema_migrations (version, inserted_at) FROM stdin;
2171      20211116024918      2022-05-03      08:35:23
2172      20211116045059      2022-05-03      08:35:23
2173      20211116050929      2022-05-03      08:35:23
2174      20211116051442      2022-05-03      08:35:23
2175      20211116212300      2022-05-03      08:35:23
2176      20211116213355      2022-05-03      08:35:23
2177      20211116213934      2022-05-03      08:35:23
2178      20211116214523      2022-05-03      08:35:23
2179      20211122062447      2022-05-03      08:35:23
2180      20211124070109      2022-05-03      08:35:23
2181      20211202204204      2022-05-03      08:35:23
2182      20211202204605      2022-05-03      08:35:23
2183      20211210212804      2022-05-03      08:35:23
2184      20211228014915      2022-05-03      08:35:23
2185      20220107221237      2022-05-03      08:35:23
2186      20220228202821      2022-05-03      08:35:23
2187      20220312004840      2022-05-03      08:35:23
2188      \.
2189
2190
2191      --
2192      -- Data for Name: subscription; Type: TABLE DATA; Schema: realtime; Owner: supabase_admin
2193      --
2194
2195      COPY realtime.subscription (id, subscription_id, entity, filters, claims, created_at)
FROM stdin;
2196      \.
2197
2198
2199      --
2200      -- Data for Name: buckets; Type: TABLE DATA; Schema: storage; Owner:
supabase_storage_admin
2201      --
2202
2203      COPY storage.buckets (id, name, owner, created_at, updated_at, public) FROM stdin;
2204      images      images      \N      2022-05-10      11:21:08.894454+00      2022-05-10      11:21:08.894454+00      t
2205      \.
2206
2207
2208      --

```

```

2209  -- Data for Name: migrations; Type: TABLE DATA; Schema: storage; Owner:
supabase_storage_admin
2210  --
2211
2212  COPY storage.migrations (id, name, hash, executed_at) FROM stdin;
2213  0    create-migrations-table e18db593bcde2aca2a408c4d1100f6abba2195df      2022-05-03 08:35
:17.497422
2214  1    initialmigration      6ab16121fbba08bbd11b712d05f358f9b555d777      2022-05-03 08:35:
17.502649
2215  2    pathtoken-column      49756be03be4c17bb85fe70d4a861f27de7e49ad      2022-05-03 08:35:
17.506822
2216  3    add-migrations-rls      bb5d124c53d68635a883e399426c6a5a25fc893d      2022-05-03 08:35:
17.530562
2217  4    add-size-functions      6d79007d04f5acd288c9c250c42d2d5fd286c54d      2022-05-03 08:35:
17.534984
2218  5    change-column-name-in-get-size fd65688505d2ffa9fbdcc58a944348dd8604d688c      2022-05-
03 08:35:17.540145
2219  6    add-rls-to-buckets      63e2bab75a2040fee8e3fb3f15a0d26f3380e9b6      2022-05-03 08:35:
17.546741
2220  7    add-public-to-buckets      82568934f8a4d9e0a85f126f6fb483ad8214c418      2022-05-03 08:35
:17.551492
2221  8    fix-search-function      1a43a40eddb525f2e2f26efd709e6c06e58e059c      2022-05-03 08:35:
17.557016
2222  9    search-files-search-function 34c096597eb8b9d077fdfdde9878c88501b2fafc      2022-05-
03 08:35:17.562544
2223  \.
2224
2225
2226  --
2227  -- Data for Name: objects; Type: TABLE DATA; Schema: storage; Owner:
supabase_storage_admin
2228  --
2229
2230  COPY storage.objects (id, bucket_id, name, owner, created_at, updated_at,
last_accessed_at, metadata) FROM stdin;
2231  1a24f118-d8cf-44c4-8820-196c45a33bd4      images .emptyFolderPlaceholder \N      2022-05-10
11:23:17.18104+00      2022-05-10 11:23:17.18104+00      2022-05-10 11:23:17.18104+00      {
"size": 0, "mimetype": "application/octet-stream", "cacheControl": "max-age=3600"}
2232  1474c152-9408-4304-a867-790ace7f11ed      images R001.jpg \N      2022-05-10 13:13:
19.084186+00      2022-05-10 13:13:19.084186+00      2022-05-10 13:13:19.084186+00      {"size":
237753, "mimetype": "image/jpeg", "cacheControl": "max-age=3600"}
2233  f019f347-4d36-475e-ba5e-3d181ea9f3d9      images R002.jpg \N      2022-05-10 13:13:
19.130669+00      2022-05-10 13:13:19.130669+00      2022-05-10 13:13:19.130669+00      {"size":
331528, "mimetype": "image/jpeg", "cacheControl": "max-age=3600"}
2234  862af812-ecb8-4f77-8c22-029a2bcd9db      images R003.jpg \N      2022-05-10 13:13:
19.259245+00      2022-05-10 13:13:19.259245+00      2022-05-10 13:13:19.259245+00      {"size":
635309, "mimetype": "image/jpeg", "cacheControl": "max-age=3600"}
2235  87136908-0fc3-4014-a087-51003199bb25      images S001.jpg \N      2022-05-10 13:13:
19.231481+00      2022-05-10 13:13:19.231481+00      2022-05-10 13:13:36.289+00      {"size":
703551, "mimetype": "image/jpeg", "cacheControl": "max-age=3600"}
2236  50a42f99-a548-4732-b2c8-bf55239dc5fa      images S002.jpg \N      2022-05-10 13:13:
19.221684+00      2022-05-10 13:13:19.221684+00      2022-05-10 13:13:43.774+00      {"size":
622772, "mimetype": "image/jpeg", "cacheControl": "max-age=3600"}
2237  17d6445a-bf45-4d36-9e33-d8a4ceb54d2f      images S003.jpg \N      2022-05-10 13:13:
19.591688+00      2022-05-10 13:13:19.591688+00      2022-05-10 13:13:50.119+00      {"size":
360941, "mimetype": "image/jpeg", "cacheControl": "max-age=3600"}
2238  0f693c7b-6377-4932-a9d7-38e64753124e      images S004.jpg \N      2022-05-10 13:13:
19.630207+00      2022-05-10 13:13:19.630207+00      2022-05-10 13:13:55.81+00      {"size":
286014, "mimetype": "image/jpeg", "cacheControl": "max-age=3600"}
2239  df408a4d-17f9-42ad-b174-3e28ab04384e      images S005.jpg \N      2022-05-14 17:39:
18.373798+00      2022-05-14 17:39:18.373798+00      2022-05-14 17:39:46.911+00      {"size":
428782, "mimetype": "image/jpeg", "cacheControl": "max-age=3600"}
2240  2f86c7eb-17e7-4c21-b469-7b04809089fc      images S006.jpg \N      2022-05-14 17:39:
26.107618+00      2022-05-14 17:39:26.107618+00      2022-05-14 17:39:54.668+00      {"size":

```



```
388363, "mimetype": "image/jpeg", "cacheControl": "max-age=3600"}
2241 e0c5d966-41df-46ab-ad51-0f1574e8d1b1 images S007.jpg \N 2022-05-14 17:39:
25.711855+00 2022-05-14 17:39:25.711855+00 2022-05-14 17:40:01.348+00 {"size":
149362, "mimetype": "image/jpeg", "cacheControl": "max-age=3600"}
2242 e6e5cd24-d5ae-4104-b489-10blac6d6575 images S008.jpg \N 2022-05-14 19:26:
52.608695+00 2022-05-14 19:26:52.608695+00 2022-05-14 19:27:12.05+00 {"size":
432611, "mimetype": "image/jpeg", "cacheControl": "max-age=3600"}
2243 8307ed92-553e-45d2-b137-02871ba93f9b images S009.jpg \N 2022-05-14 19:26:
52.328964+00 2022-05-14 19:26:52.328964+00 2022-05-14 19:27:26.024+00 {"size":
186799, "mimetype": "image/jpeg", "cacheControl": "max-age=3600"}
2244 \.
2245
2246
2247 --
2248 -- Name: refresh_tokens_id_seq; Type: SEQUENCE SET; Schema: auth; Owner:
supabase_auth_admin
2249 --
2250
2251 SELECT pg_catalog.setval('auth.refresh_tokens_id_seq', 1, false);
2252
2253
2254 --
2255 -- Name: Decision_id_seq; Type: SEQUENCE SET; Schema: public; Owner: supabase_admin
2256 --
2257
2258 SELECT pg_catalog.setval('public."Decision_id_seq"', 1, true);
2259
2260
2261 --
2262 -- Name: Donation_id_seq; Type: SEQUENCE SET; Schema: public; Owner: supabase_admin
2263 --
2264
2265 SELECT pg_catalog.setval('public."Donation_id_seq"', 1, false);
2266
2267
2268 --
2269 -- Name: Donor_id_seq; Type: SEQUENCE SET; Schema: public; Owner: supabase_admin
2270 --
2271
2272 SELECT pg_catalog.setval('public."Donor_id_seq"', 1, false);
2273
2274
2275 --
2276 -- Name: Governors_id_seq; Type: SEQUENCE SET; Schema: public; Owner: supabase_admin
2277 --
2278
2279 SELECT pg_catalog.setval('public."Governors_id_seq"', 1, false);
2280
2281
2282 --
2283 -- Name: Organizations_id_seq; Type: SEQUENCE SET; Schema: public; Owner: supabase_admin
2284 --
2285
2286 SELECT pg_catalog.setval('public."Organizations_id_seq"', 4, true);
2287
2288
2289 --
2290 -- Name: Projects_id_seq; Type: SEQUENCE SET; Schema: public; Owner: supabase_admin
2291 --
2292
2293 SELECT pg_catalog.setval('public."Projects_id_seq"', 6, true);
2294
2295
2296 --
```



```
2297 -- Name: SingleVote_id_seq; Type: SEQUENCE SET; Schema: public; Owner: supabase_admin
2298 --
2299
2300 SELECT pg_catalog.setval('public."SingleVote_id_seq"', 1, false);
2301
2302
2303 --
2304 -- Name: Solutions_id_seq; Type: SEQUENCE SET; Schema: public; Owner: supabase_admin
2305 --
2306
2307 SELECT pg_catalog.setval('public."Solutions_id_seq"', 1, false);
2308
2309
2310 --
2311 -- Name: SystemConstants_id_seq; Type: SEQUENCE SET; Schema: public; Owner:
supabase_admin
2312 --
2313
2314 SELECT pg_catalog.setval('public."SystemConstants_id_seq"', 1, false);
2315
2316
2317 --
2318 -- Name: subscription_id_seq; Type: SEQUENCE SET; Schema: realtime; Owner: supabase_admin
2319 --
2320
2321 SELECT pg_catalog.setval('realtime.subscription_id_seq', 1, false);
2322
2323
2324 --
2325 -- Name: audit_log_entries audit_log_entries_pkey; Type: CONSTRAINT; Schema: auth;
Owner: supabase_auth_admin
2326 --
2327
2328 ALTER TABLE ONLY auth.audit_log_entries
2329     ADD CONSTRAINT audit_log_entries_pkey PRIMARY KEY (id);
2330
2331
2332 --
2333 -- Name: identities identities_pkey; Type: CONSTRAINT; Schema: auth; Owner:
supabase_auth_admin
2334 --
2335
2336 ALTER TABLE ONLY auth.identities
2337     ADD CONSTRAINT identities_pkey PRIMARY KEY (provider, id);
2338
2339
2340 --
2341 -- Name: instances instances_pkey; Type: CONSTRAINT; Schema: auth; Owner:
supabase_auth_admin
2342 --
2343
2344 ALTER TABLE ONLY auth.instances
2345     ADD CONSTRAINT instances_pkey PRIMARY KEY (id);
2346
2347
2348 --
2349 -- Name: refresh_tokens refresh_tokens_pkey; Type: CONSTRAINT; Schema: auth; Owner:
supabase_auth_admin
2350 --
2351
2352 ALTER TABLE ONLY auth.refresh_tokens
2353     ADD CONSTRAINT refresh_tokens_pkey PRIMARY KEY (id);
2354
2355
```

```
2356  --
2357  -- Name: refresh_tokens refresh_tokens_token_unique; Type: CONSTRAINT; Schema: auth;
Owner: supabase_auth_admin
2358  --
2359
2360  ALTER TABLE ONLY auth.refresh_tokens
2361      ADD CONSTRAINT refresh_tokens_token_unique UNIQUE (token);
2362
2363
2364  --
2365  -- Name: schema_migrations schema_migrations_pkey; Type: CONSTRAINT; Schema: auth;
Owner: supabase_auth_admin
2366  --
2367
2368  ALTER TABLE ONLY auth.schema_migrations
2369      ADD CONSTRAINT schema_migrations_pkey PRIMARY KEY (version);
2370
2371
2372  --
2373  -- Name: users users_email_key; Type: CONSTRAINT; Schema: auth; Owner:
supabase_auth_admin
2374  --
2375
2376  ALTER TABLE ONLY auth.users
2377      ADD CONSTRAINT users_email_key UNIQUE (email);
2378
2379
2380  --
2381  -- Name: users users_phone_key; Type: CONSTRAINT; Schema: auth; Owner:
supabase_auth_admin
2382  --
2383
2384  ALTER TABLE ONLY auth.users
2385      ADD CONSTRAINT users_phone_key UNIQUE (phone);
2386
2387
2388  --
2389  -- Name: users users_pkey; Type: CONSTRAINT; Schema: auth; Owner: supabase_auth_admin
2390  --
2391
2392  ALTER TABLE ONLY auth.users
2393      ADD CONSTRAINT users_pkey PRIMARY KEY (id);
2394
2395
2396  --
2397  -- Name: Decisions Decision_pkey; Type: CONSTRAINT; Schema: public; Owner: supabase_admin
2398  --
2399
2400  ALTER TABLE ONLY public."Decisions"
2401      ADD CONSTRAINT "Decision_pkey" PRIMARY KEY (id);
2402
2403
2404  --
2405  -- Name: Donations Donation_pkey; Type: CONSTRAINT; Schema: public; Owner: supabase_admin
2406  --
2407
2408  ALTER TABLE ONLY public."Donations"
2409      ADD CONSTRAINT "Donation_pkey" PRIMARY KEY (id);
2410
2411
2412  --
2413  -- Name: Donors Donor_pkey; Type: CONSTRAINT; Schema: public; Owner: supabase_admin
2414  --
2415
```

```
2416 ALTER TABLE ONLY public."Donors"
2417     ADD CONSTRAINT "Donor_pkey" PRIMARY KEY (id);
2418
2419
2420 --
2421 -- Name: Governors Governors_pkey; Type: CONSTRAINT; Schema: public; Owner:
supabase_admin
2422 --
2423
2424 ALTER TABLE ONLY public."Governors"
2425     ADD CONSTRAINT "Governors_pkey" PRIMARY KEY (id);
2426
2427
2428 --
2429 -- Name: Organizations Organizations_pkey; Type: CONSTRAINT; Schema: public; Owner:
supabase_admin
2430 --
2431
2432 ALTER TABLE ONLY public."Organizations"
2433     ADD CONSTRAINT "Organizations_pkey" PRIMARY KEY (id);
2434
2435
2436 --
2437 -- Name: Projects Projects_pkey; Type: CONSTRAINT; Schema: public; Owner: supabase_admin
2438 --
2439
2440 ALTER TABLE ONLY public."Projects"
2441     ADD CONSTRAINT "Projects_pkey" PRIMARY KEY (id);
2442
2443
2444 --
2445 -- Name: SingleVotes SingleVote_pkey; Type: CONSTRAINT; Schema: public; Owner:
supabase_admin
2446 --
2447
2448 ALTER TABLE ONLY public."SingleVotes"
2449     ADD CONSTRAINT "SingleVote_pkey" PRIMARY KEY (id);
2450
2451
2452 --
2453 -- Name: Solutions Solutions_pkey; Type: CONSTRAINT; Schema: public; Owner:
supabase_admin
2454 --
2455
2456 ALTER TABLE ONLY public."Solutions"
2457     ADD CONSTRAINT "Solutions_pkey" PRIMARY KEY (id);
2458
2459
2460 --
2461 -- Name: SystemConstants SystemConstants_pkey; Type: CONSTRAINT; Schema: public; Owner:
supabase_admin
2462 --
2463
2464 ALTER TABLE ONLY public."SystemConstants"
2465     ADD CONSTRAINT "SystemConstants_pkey" PRIMARY KEY (id);
2466
2467
2468 --
2469 -- Name: subscription pk_subscription; Type: CONSTRAINT; Schema: realtime; Owner:
supabase_admin
2470 --
2471
2472 ALTER TABLE ONLY realtime.subscription
2473     ADD CONSTRAINT pk_subscription PRIMARY KEY (id);
```

```
2474
2475
2476 --
2477 -- Name: schema_migrations schema_migrations_pkey; Type: CONSTRAINT; Schema: realtime;
Owner: supabase_admin
2478 --
2479
2480 ALTER TABLE ONLY realtime.schema_migrations
2481     ADD CONSTRAINT schema_migrations_pkey PRIMARY KEY (version);
2482
2483
2484 --
2485 -- Name: buckets buckets_pkey; Type: CONSTRAINT; Schema: storage; Owner:
supabase_storage_admin
2486 --
2487
2488 ALTER TABLE ONLY storage.buckets
2489     ADD CONSTRAINT buckets_pkey PRIMARY KEY (id);
2490
2491
2492 --
2493 -- Name: migrations migrations_name_key; Type: CONSTRAINT; Schema: storage; Owner:
supabase_storage_admin
2494 --
2495
2496 ALTER TABLE ONLY storage.migrations
2497     ADD CONSTRAINT migrations_name_key UNIQUE (name);
2498
2499
2500 --
2501 -- Name: migrations migrations_pkey; Type: CONSTRAINT; Schema: storage; Owner:
supabase_storage_admin
2502 --
2503
2504 ALTER TABLE ONLY storage.migrations
2505     ADD CONSTRAINT migrations_pkey PRIMARY KEY (id);
2506
2507
2508 --
2509 -- Name: objects objects_pkey; Type: CONSTRAINT; Schema: storage; Owner:
supabase_storage_admin
2510 --
2511
2512 ALTER TABLE ONLY storage.objects
2513     ADD CONSTRAINT objects_pkey PRIMARY KEY (id);
2514
2515
2516 --
2517 -- Name: audit_logs_instance_id_idx; Type: INDEX; Schema: auth; Owner:
supabase_auth_admin
2518 --
2519
2520 CREATE INDEX audit_logs_instance_id_idx ON auth.audit_log_entries USING btree (
instance_id);
2521
2522
2523 --
2524 -- Name: confirmation_token_idx; Type: INDEX; Schema: auth; Owner: supabase_auth_admin
2525 --
2526
2527 CREATE UNIQUE INDEX confirmation_token_idx ON auth.users USING btree (confirmation_token
) WHERE ((confirmation_token)::text !~ '^[0-9 ]*$'::text);
2528
2529
```

```
2530  --
2531  -- Name: email_change_token_current_idx; Type: INDEX; Schema: auth; Owner:
supabase_auth_admin
2532  --
2533
2534  CREATE UNIQUE INDEX email_change_token_current_idx ON auth.users USING btree (
email_change_token_current) WHERE ((email_change_token_current)::text !~ '^[0-9 ]*$'::
text);
2535
2536
2537  --
2538  -- Name: email_change_token_new_idx; Type: INDEX; Schema: auth; Owner:
supabase_auth_admin
2539  --
2540
2541  CREATE UNIQUE INDEX email_change_token_new_idx ON auth.users USING btree (
email_change_token_new) WHERE ((email_change_token_new)::text !~ '^[0-9 ]*$'::text);
2542
2543
2544  --
2545  -- Name: identities_user_id_idx; Type: INDEX; Schema: auth; Owner: supabase_auth_admin
2546  --
2547
2548  CREATE INDEX identities_user_id_idx ON auth.identities USING btree (user_id);
2549
2550
2551  --
2552  -- Name: reauthentication_token_idx; Type: INDEX; Schema: auth; Owner:
supabase_auth_admin
2553  --
2554
2555  CREATE UNIQUE INDEX reauthentication_token_idx ON auth.users USING btree (
reauthentication_token) WHERE ((reauthentication_token)::text !~ '^[0-9 ]*$'::text);
2556
2557
2558  --
2559  -- Name: recovery_token_idx; Type: INDEX; Schema: auth; Owner: supabase_auth_admin
2560  --
2561
2562  CREATE UNIQUE INDEX recovery_token_idx ON auth.users USING btree (recovery_token) WHERE
((recovery_token)::text !~ '^[0-9 ]*$'::text);
2563
2564
2565  --
2566  -- Name: refresh_tokens_instance_id_idx; Type: INDEX; Schema: auth; Owner:
supabase_auth_admin
2567  --
2568
2569  CREATE INDEX refresh_tokens_instance_id_idx ON auth.refresh_tokens USING btree (
instance_id);
2570
2571
2572  --
2573  -- Name: refresh_tokens_instance_id_user_id_idx; Type: INDEX; Schema: auth; Owner:
supabase_auth_admin
2574  --
2575
2576  CREATE INDEX refresh_tokens_instance_id_user_id_idx ON auth.refresh_tokens USING btree (
instance_id, user_id);
2577
2578
2579  --
2580  -- Name: refresh_tokens_parent_idx; Type: INDEX; Schema: auth; Owner: supabase_auth_admin
2581  --
```

```
2582
2583 CREATE INDEX refresh_tokens_parent_idx ON auth.refresh_tokens USING btree (parent);
2584
2585
2586 --
2587 -- Name: refresh_tokens_token_idx; Type: INDEX; Schema: auth; Owner: supabase_auth_admin
2588 --
2589
2590 CREATE INDEX refresh_tokens_token_idx ON auth.refresh_tokens USING btree (token);
2591
2592
2593 --
2594 -- Name: users_instance_id_email_idx; Type: INDEX; Schema: auth; Owner:
supabase_auth_admin
2595 --
2596
2597 CREATE INDEX users_instance_id_email_idx ON auth.users USING btree (instance_id, lower((
email)::text));
2598
2599
2600 --
2601 -- Name: users_instance_id_idx; Type: INDEX; Schema: auth; Owner: supabase_auth_admin
2602 --
2603
2604 CREATE INDEX users_instance_id_idx ON auth.users USING btree (instance_id);
2605
2606
2607 --
2608 -- Name: ix_realtime_subscription_entity; Type: INDEX; Schema: realtime; Owner:
supabase_admin
2609 --
2610
2611 CREATE INDEX ix_realtime_subscription_entity ON realtime.subscription USING hash (entity
);
2612
2613
2614 --
2615 -- Name: subscription_subscription_id_entity_filters_key; Type: INDEX; Schema:
realtime; Owner: supabase_admin
2616 --
2617
2618 CREATE UNIQUE INDEX subscription_subscription_id_entity_filters_key ON realtime.
subscription USING btree (subscription_id, entity, filters);
2619
2620
2621 --
2622 -- Name: bname; Type: INDEX; Schema: storage; Owner: supabase_storage_admin
2623 --
2624
2625 CREATE UNIQUE INDEX bname ON storage.buckets USING btree (name);
2626
2627
2628 --
2629 -- Name: bucketid_objname; Type: INDEX; Schema: storage; Owner: supabase_storage_admin
2630 --
2631
2632 CREATE UNIQUE INDEX bucketid_objname ON storage.objects USING btree (bucket_id, name);
2633
2634
2635 --
2636 -- Name: name_prefix_search; Type: INDEX; Schema: storage; Owner: supabase_storage_admin
2637 --
2638
2639 CREATE INDEX name_prefix_search ON storage.objects USING btree (name text_pattern_ops);
```

```

2640
2641
2642 --
2643 -- Name: subscription tr_check_filters; Type: TRIGGER; Schema: realtime; Owner:
supabase_admin
2644 --
2645
2646 CREATE TRIGGER tr_check_filters BEFORE INSERT OR UPDATE ON realtime.subscription FOR
EACH ROW EXECUTE FUNCTION realtime.subscription_check_filters();
2647
2648
2649 --
2650 -- Name: identities identities_user_id_fkey; Type: FK CONSTRAINT; Schema: auth; Owner:
supabase_auth_admin
2651 --
2652
2653 ALTER TABLE ONLY auth.identities
2654     ADD CONSTRAINT identities_user_id_fkey FOREIGN KEY (user_id) REFERENCES auth.users(
id) ON DELETE CASCADE;
2655
2656
2657 --
2658 -- Name: refresh_tokens refresh_tokens_parent_fkey; Type: FK CONSTRAINT; Schema: auth;
Owner: supabase_auth_admin
2659 --
2660
2661 ALTER TABLE ONLY auth.refresh_tokens
2662     ADD CONSTRAINT refresh_tokens_parent_fkey FOREIGN KEY (parent) REFERENCES auth.
refresh_tokens(token);
2663
2664
2665 --
2666 -- Name: Donations Donation_donor_id_fkey; Type: FK CONSTRAINT; Schema: public; Owner:
supabase_admin
2667 --
2668
2669 ALTER TABLE ONLY public."Donations"
2670     ADD CONSTRAINT "Donation_donor_id_fkey" FOREIGN KEY (donor_id) REFERENCES public.
"Donors"(id);
2671
2672
2673 --
2674 -- Name: Donations Donation_project_id_fkey; Type: FK CONSTRAINT; Schema: public;
Owner: supabase_admin
2675 --
2676
2677 ALTER TABLE ONLY public."Donations"
2678     ADD CONSTRAINT "Donation_project_id_fkey" FOREIGN KEY (project_id) REFERENCES public
."Projects"(id);
2679
2680
2681 --
2682 -- Name: Projects Projects_organization_id_fkey; Type: FK CONSTRAINT; Schema: public;
Owner: supabase_admin
2683 --
2684
2685 ALTER TABLE ONLY public."Projects"
2686     ADD CONSTRAINT "Projects_organization_id_fkey" FOREIGN KEY (organization_id)
REFERENCES public."Organizations"(id);
2687
2688
2689 --
2690 -- Name: Projects Projects_solution_id_fkey; Type: FK CONSTRAINT; Schema: public;
Owner: supabase_admin

```

```
2691  --
2692
2693  ALTER TABLE ONLY public."Projects"
2694      ADD CONSTRAINT "Projects_solution_id_fkey" FOREIGN KEY (solution_id) REFERENCES
        public."Solutions"(id);
2695
2696
2697  --
2698  -- Name: SingleVotes SingleVote_governor_id_fkey; Type: FK CONSTRAINT; Schema: public;
Owner: supabase_admin
2699  --
2700
2701  ALTER TABLE ONLY public."SingleVotes"
2702      ADD CONSTRAINT "SingleVote_governor_id_fkey" FOREIGN KEY (governor_id) REFERENCES
        public."Governors"(id);
2703
2704
2705  --
2706  -- Name: SingleVotes SingleVotes_project_id_fkey; Type: FK CONSTRAINT; Schema: public;
Owner: supabase_admin
2707  --
2708
2709  ALTER TABLE ONLY public."SingleVotes"
2710      ADD CONSTRAINT "SingleVotes_project_id_fkey" FOREIGN KEY (project_id) REFERENCES
        public."Projects"(id);
2711
2712
2713  --
2714  -- Name: buckets buckets_owner_fkey; Type: FK CONSTRAINT; Schema: storage; Owner:
supabase_storage_admin
2715  --
2716
2717  ALTER TABLE ONLY storage.buckets
2718      ADD CONSTRAINT buckets_owner_fkey FOREIGN KEY (owner) REFERENCES auth.users(id);
2719
2720
2721  --
2722  -- Name: objects objects_bucketId_fkey; Type: FK CONSTRAINT; Schema: storage; Owner:
supabase_storage_admin
2723  --
2724
2725  ALTER TABLE ONLY storage.objects
2726      ADD CONSTRAINT "objects_bucketId_fkey" FOREIGN KEY (bucket_id) REFERENCES storage.
        buckets(id);
2727
2728
2729  --
2730  -- Name: objects objects_owner_fkey; Type: FK CONSTRAINT; Schema: storage; Owner:
supabase_storage_admin
2731  --
2732
2733  ALTER TABLE ONLY storage.objects
2734      ADD CONSTRAINT objects_owner_fkey FOREIGN KEY (owner) REFERENCES auth.users(id);
2735
2736
2737  --
2738  -- Name: buckets; Type: ROW SECURITY; Schema: storage; Owner: supabase_storage_admin
2739  --
2740
2741  ALTER TABLE storage.buckets ENABLE ROW LEVEL SECURITY;
2742
2743  --
2744  -- Name: migrations; Type: ROW SECURITY; Schema: storage; Owner: supabase_storage_admin
2745  --
```



```
2746
2747 ALTER TABLE storage.migrations ENABLE ROW LEVEL SECURITY;
2748
2749 --
2750 -- Name: objects; Type: ROW SECURITY; Schema: storage; Owner: supabase_storage_admin
2751 --
2752
2753 ALTER TABLE storage.objects ENABLE ROW LEVEL SECURITY;
2754
2755 --
2756 -- Name: supabase_realtime; Type: PUBLICATION; Schema: -; Owner: postgres
2757 --
2758
2759 CREATE PUBLICATION supabase_realtime WITH (publish = 'insert, update, delete, truncate');
2760
2761
2762 ALTER PUBLICATION supabase_realtime OWNER TO postgres;
2763
2764 --
2765 -- Name: SCHEMA auth; Type: ACL; Schema: -; Owner: supabase_admin
2766 --
2767
2768 GRANT USAGE ON SCHEMA auth TO anon;
2769 GRANT USAGE ON SCHEMA auth TO authenticated;
2770 GRANT USAGE ON SCHEMA auth TO service_role;
2771 GRANT ALL ON SCHEMA auth TO supabase_auth_admin;
2772 GRANT ALL ON SCHEMA auth TO dashboard_user;
2773 GRANT ALL ON SCHEMA auth TO postgres;
2774
2775
2776 --
2777 -- Name: SCHEMA extensions; Type: ACL; Schema: -; Owner: postgres
2778 --
2779
2780 GRANT USAGE ON SCHEMA extensions TO anon;
2781 GRANT USAGE ON SCHEMA extensions TO authenticated;
2782 GRANT USAGE ON SCHEMA extensions TO service_role;
2783 GRANT ALL ON SCHEMA extensions TO dashboard_user;
2784
2785
2786 --
2787 -- Name: SCHEMA public; Type: ACL; Schema: -; Owner: postgres
2788 --
2789
2790 GRANT USAGE ON SCHEMA public TO anon;
2791 GRANT USAGE ON SCHEMA public TO authenticated;
2792 GRANT USAGE ON SCHEMA public TO service_role;
2793
2794
2795 --
2796 -- Name: SCHEMA graphql_public; Type: ACL; Schema: -; Owner: supabase_admin
2797 --
2798
2799 GRANT USAGE ON SCHEMA graphql_public TO postgres;
2800 GRANT USAGE ON SCHEMA graphql_public TO anon;
2801 GRANT USAGE ON SCHEMA graphql_public TO authenticated;
2802 GRANT USAGE ON SCHEMA graphql_public TO service_role;
2803
2804
2805 --
2806 -- Name: SCHEMA realtime; Type: ACL; Schema: -; Owner: supabase_admin
2807 --
2808
2809 GRANT USAGE ON SCHEMA realtime TO postgres;
```

```
2810
2811
2812 --
2813 -- Name: SCHEMA storage; Type: ACL; Schema: -; Owner: supabase_admin
2814 --
2815
2816 GRANT ALL ON SCHEMA storage TO postgres;
2817 GRANT USAGE ON SCHEMA storage TO anon;
2818 GRANT USAGE ON SCHEMA storage TO authenticated;
2819 GRANT USAGE ON SCHEMA storage TO service_role;
2820 GRANT ALL ON SCHEMA storage TO supabase_storage_admin;
2821 GRANT ALL ON SCHEMA storage TO dashboard_user;
2822
2823
2824 --
2825 -- Name: FUNCTION email(); Type: ACL; Schema: auth; Owner: supabase_auth_admin
2826 --
2827
2828 GRANT ALL ON FUNCTION auth.email() TO dashboard_user;
2829
2830
2831 --
2832 -- Name: FUNCTION role(); Type: ACL; Schema: auth; Owner: supabase_auth_admin
2833 --
2834
2835 GRANT ALL ON FUNCTION auth.role() TO dashboard_user;
2836
2837
2838 --
2839 -- Name: FUNCTION uid(); Type: ACL; Schema: auth; Owner: supabase_auth_admin
2840 --
2841
2842 GRANT ALL ON FUNCTION auth.uid() TO dashboard_user;
2843
2844
2845 --
2846 -- Name: FUNCTION algorithm_sign(signables text, secret text, algorithm text); Type:
ACL; Schema: extensions; Owner: postgres
2847 --
2848
2849 GRANT ALL ON FUNCTION extensions.algorithm_sign(signables text, secret text, algorithm
text) TO dashboard_user;
2850
2851
2852 --
2853 -- Name: FUNCTION armor(bytea); Type: ACL; Schema: extensions; Owner: postgres
2854 --
2855
2856 GRANT ALL ON FUNCTION extensions.armor(bytea) TO dashboard_user;
2857
2858
2859 --
2860 -- Name: FUNCTION armor(bytea, text[], text[]); Type: ACL; Schema: extensions; Owner:
postgres
2861 --
2862
2863 GRANT ALL ON FUNCTION extensions.armor(bytea, text[], text[]) TO dashboard_user;
2864
2865
2866 --
2867 -- Name: FUNCTION crypt(text, text); Type: ACL; Schema: extensions; Owner: postgres
2868 --
2869
2870 GRANT ALL ON FUNCTION extensions.crypt(text, text) TO dashboard_user;
```

```
2871
2872
2873 --
2874 -- Name: FUNCTION dearmor(text); Type: ACL; Schema: extensions; Owner: postgres
2875 --
2876
2877 GRANT ALL ON FUNCTION extensions.dearmor(text) TO dashboard_user;
2878
2879
2880 --
2881 -- Name: FUNCTION decrypt(bytea, bytea, text); Type: ACL; Schema: extensions; Owner:
postgres
2882 --
2883
2884 GRANT ALL ON FUNCTION extensions.decrypt(bytea, bytea, text) TO dashboard_user;
2885
2886
2887 --
2888 -- Name: FUNCTION decrypt_iv(bytea, bytea, bytea, text); Type: ACL; Schema: extensions;
Owner: postgres
2889 --
2890
2891 GRANT ALL ON FUNCTION extensions.decrypt_iv(bytea, bytea, bytea, text) TO dashboard_user;
2892
2893
2894 --
2895 -- Name: FUNCTION digest(bytea, text); Type: ACL; Schema: extensions; Owner: postgres
2896 --
2897
2898 GRANT ALL ON FUNCTION extensions.digest(bytea, text) TO dashboard_user;
2899
2900
2901 --
2902 -- Name: FUNCTION digest(text, text); Type: ACL; Schema: extensions; Owner: postgres
2903 --
2904
2905 GRANT ALL ON FUNCTION extensions.digest(text, text) TO dashboard_user;
2906
2907
2908 --
2909 -- Name: FUNCTION encrypt(bytea, bytea, text); Type: ACL; Schema: extensions; Owner:
postgres
2910 --
2911
2912 GRANT ALL ON FUNCTION extensions.encrypt(bytea, bytea, text) TO dashboard_user;
2913
2914
2915 --
2916 -- Name: FUNCTION encrypt_iv(bytea, bytea, bytea, text); Type: ACL; Schema: extensions;
Owner: postgres
2917 --
2918
2919 GRANT ALL ON FUNCTION extensions.encrypt_iv(bytea, bytea, bytea, text) TO dashboard_user;
2920
2921
2922 --
2923 -- Name: FUNCTION gen_random_bytes(integer); Type: ACL; Schema: extensions; Owner:
postgres
2924 --
2925
2926 GRANT ALL ON FUNCTION extensions.gen_random_bytes(integer) TO dashboard_user;
2927
2928
2929 --
```

```

2930  -- Name: FUNCTION gen_random_uuid(); Type: ACL; Schema: extensions; Owner: postgres
2931  --
2932
2933  GRANT ALL ON FUNCTION extensions.gen_random_uuid() TO dashboard_user;
2934
2935
2936  --
2937  -- Name: FUNCTION gen_salt(text); Type: ACL; Schema: extensions; Owner: postgres
2938  --
2939
2940  GRANT ALL ON FUNCTION extensions.gen_salt(text) TO dashboard_user;
2941
2942
2943  --
2944  -- Name: FUNCTION gen_salt(text, integer); Type: ACL; Schema: extensions; Owner: postgres
2945  --
2946
2947  GRANT ALL ON FUNCTION extensions.gen_salt(text, integer) TO dashboard_user;
2948
2949
2950  --
2951  -- Name: FUNCTION grant_pg_cron_access(); Type: ACL; Schema: extensions; Owner: postgres
2952  --
2953
2954  GRANT ALL ON FUNCTION extensions.grant_pg_cron_access() TO dashboard_user;
2955
2956
2957  --
2958  -- Name: FUNCTION grant_pg_net_access(); Type: ACL; Schema: extensions; Owner: postgres
2959  --
2960
2961  GRANT ALL ON FUNCTION extensions.grant_pg_net_access() TO dashboard_user;
2962
2963
2964  --
2965  -- Name: FUNCTION hmac(bytea, bytea, text); Type: ACL; Schema: extensions; Owner:
postgres
2966  --
2967
2968  GRANT ALL ON FUNCTION extensions.hmac(bytea, bytea, text) TO dashboard_user;
2969
2970
2971  --
2972  -- Name: FUNCTION hmac(text, text, text); Type: ACL; Schema: extensions; Owner: postgres
2973  --
2974
2975  GRANT ALL ON FUNCTION extensions.hmac(text, text, text) TO dashboard_user;
2976
2977
2978  --
2979  -- Name: FUNCTION pg_stat_statements(showtext boolean, OUT userid oid, OUT dbid oid,
OUT toplevel boolean, OUT queryid bigint, OUT query text, OUT plans bigint, OUT
total_plan_time double precision, OUT min_plan_time double precision, OUT max_plan_time
double precision, OUT mean_plan_time double precision, OUT stddev_plan_time double
precision, OUT calls bigint, OUT total_exec_time double precision, OUT min_exec_time
double precision, OUT max_exec_time double precision, OUT mean_exec_time double
precision, OUT stddev_exec_time double precision, OUT rows bigint, OUT shared_blks_hit
bigint, OUT shared_blks_read bigint, OUT shared_blks_dirtied bigint, OUT
shared_blks_written bigint, OUT local_blks_hit bigint, OUT local_blks_read bigint, OUT
local_blks_dirtied bigint, OUT local_blks_written bigint, OUT temp_blks_read bigint,
OUT temp_blks_written bigint, OUT blk_read_time double precision, OUT blk_write_time
double precision, OUT wal_records bigint, OUT wal_fpi bigint, OUT wal_bytes numeric);
Type: ACL; Schema: extensions; Owner: postgres
2980  --

```

```

2981
2982 GRANT ALL ON FUNCTION extensions.pg_stat_statements(showtext boolean, OUT userid oid,
OUT dbid oid, OUT toplevel boolean, OUT queryid bigint, OUT query text, OUT plans bigint
, OUT total_plan_time double precision, OUT min_plan_time double precision, OUT
max_plan_time double precision, OUT mean_plan_time double precision, OUT
stddev_plan_time double precision, OUT calls bigint, OUT total_exec_time double
precision, OUT min_exec_time double precision, OUT max_exec_time double precision, OUT
mean_exec_time double precision, OUT stddev_exec_time double precision, OUT rows bigint,
OUT shared_blks_hit bigint, OUT shared_blks_read bigint, OUT shared_blks_dirtied bigint
, OUT shared_blks_written bigint, OUT local_blks_hit bigint, OUT local_blks_read bigint,
OUT local_blks_dirtied bigint, OUT local_blks_written bigint, OUT temp_blks_read bigint
, OUT temp_blks_written bigint, OUT blk_read_time double precision, OUT blk_write_time
double precision, OUT wal_records bigint, OUT wal_fpi bigint, OUT wal_bytes numeric) TO
dashboard_user;

2983
2984
2985 --
2986 -- Name: FUNCTION pg_stat_statements_info(OUT dealloc bigint, OUT stats_reset timestamp
with time zone); Type: ACL; Schema: extensions; Owner: postgres
2987 --
2988
2989 GRANT ALL ON FUNCTION extensions.pg_stat_statements_info(OUT dealloc bigint, OUT
stats_reset timestamp with time zone) TO dashboard_user;

2990
2991
2992 --
2993 -- Name: FUNCTION pg_stat_statements_reset(userid oid, dbid oid, queryid bigint); Type:
ACL; Schema: extensions; Owner: postgres
2994 --
2995
2996 GRANT ALL ON FUNCTION extensions.pg_stat_statements_reset(userid oid, dbid oid, queryid
bigint) TO dashboard_user;

2997
2998
2999 --
3000 -- Name: FUNCTION pgp_armor_headers(text, OUT key text, OUT value text); Type: ACL;
Schema: extensions; Owner: postgres
3001 --
3002
3003 GRANT ALL ON FUNCTION extensions.pgp_armor_headers(text, OUT key text, OUT value text)
TO dashboard_user;

3004
3005
3006 --
3007 -- Name: FUNCTION pgp_key_id(bytea); Type: ACL; Schema: extensions; Owner: postgres
3008 --
3009
3010 GRANT ALL ON FUNCTION extensions.pgp_key_id(bytea) TO dashboard_user;

3011
3012
3013 --
3014 -- Name: FUNCTION pgp_pub_decrypt(bytea, bytea); Type: ACL; Schema: extensions; Owner:
postgres
3015 --
3016
3017 GRANT ALL ON FUNCTION extensions.pgp_pub_decrypt(bytea, bytea) TO dashboard_user;

3018
3019
3020 --
3021 -- Name: FUNCTION pgp_pub_decrypt(bytea, bytea, text); Type: ACL; Schema: extensions;
Owner: postgres
3022 --
3023
3024 GRANT ALL ON FUNCTION extensions.pgp_pub_decrypt(bytea, bytea, text) TO dashboard_user;

```

```
3025
3026
3027 --
3028 -- Name: FUNCTION pgp_pub_decrypt(bytea, bytea, text, text); Type: ACL; Schema:
extensions; Owner: postgres
3029 --
3030
3031 GRANT ALL ON FUNCTION extensions.pgp_pub_decrypt(bytea, bytea, text, text) TO
dashboard_user;
3032
3033
3034 --
3035 -- Name: FUNCTION pgp_pub_decrypt_bytea(bytea, bytea); Type: ACL; Schema: extensions;
Owner: postgres
3036 --
3037
3038 GRANT ALL ON FUNCTION extensions.pgp_pub_decrypt_bytea(bytea, bytea) TO dashboard_user;
3039
3040
3041 --
3042 -- Name: FUNCTION pgp_pub_decrypt_bytea(bytea, bytea, text); Type: ACL; Schema:
extensions; Owner: postgres
3043 --
3044
3045 GRANT ALL ON FUNCTION extensions.pgp_pub_decrypt_bytea(bytea, bytea, text) TO
dashboard_user;
3046
3047
3048 --
3049 -- Name: FUNCTION pgp_pub_decrypt_bytea(bytea, bytea, text, text); Type: ACL; Schema:
extensions; Owner: postgres
3050 --
3051
3052 GRANT ALL ON FUNCTION extensions.pgp_pub_decrypt_bytea(bytea, bytea, text, text) TO
dashboard_user;
3053
3054
3055 --
3056 -- Name: FUNCTION pgp_pub_encrypt(text, bytea); Type: ACL; Schema: extensions; Owner:
postgres
3057 --
3058
3059 GRANT ALL ON FUNCTION extensions.pgp_pub_encrypt(text, bytea) TO dashboard_user;
3060
3061
3062 --
3063 -- Name: FUNCTION pgp_pub_encrypt(text, bytea, text); Type: ACL; Schema: extensions;
Owner: postgres
3064 --
3065
3066 GRANT ALL ON FUNCTION extensions.pgp_pub_encrypt(text, bytea, text) TO dashboard_user;
3067
3068
3069 --
3070 -- Name: FUNCTION pgp_pub_encrypt_bytea(bytea, bytea); Type: ACL; Schema: extensions;
Owner: postgres
3071 --
3072
3073 GRANT ALL ON FUNCTION extensions.pgp_pub_encrypt_bytea(bytea, bytea) TO dashboard_user;
3074
3075
3076 --
3077 -- Name: FUNCTION pgp_pub_encrypt_bytea(bytea, bytea, text); Type: ACL; Schema:
extensions; Owner: postgres
```

```
3078  --
3079
3080  GRANT ALL ON FUNCTION extensions.pgp_pub_encrypt_bytea(bytea, bytea, text) TO
dashboard_user;
3081
3082  --
3083  --
3084  -- Name: FUNCTION pgp_sym_decrypt(bytea, text); Type: ACL; Schema: extensions; Owner:
postgres
3085  --
3086
3087  GRANT ALL ON FUNCTION extensions.pgp_sym_decrypt(bytea, text) TO dashboard_user;
3088
3089  --
3090  --
3091  -- Name: FUNCTION pgp_sym_decrypt(bytea, text, text); Type: ACL; Schema: extensions;
Owner: postgres
3092  --
3093
3094  GRANT ALL ON FUNCTION extensions.pgp_sym_decrypt(bytea, text, text) TO dashboard_user;
3095
3096  --
3097  --
3098  -- Name: FUNCTION pgp_sym_decrypt_bytea(bytea, text); Type: ACL; Schema: extensions;
Owner: postgres
3099  --
3100
3101  GRANT ALL ON FUNCTION extensions.pgp_sym_decrypt_bytea(bytea, text) TO dashboard_user;
3102
3103  --
3104  --
3105  -- Name: FUNCTION pgp_sym_decrypt_bytea(bytea, text, text); Type: ACL; Schema:
extensions; Owner: postgres
3106  --
3107
3108  GRANT ALL ON FUNCTION extensions.pgp_sym_decrypt_bytea(bytea, text, text) TO
dashboard_user;
3109
3110  --
3111  --
3112  -- Name: FUNCTION pgp_sym_encrypt(text, text); Type: ACL; Schema: extensions; Owner:
postgres
3113  --
3114
3115  GRANT ALL ON FUNCTION extensions.pgp_sym_encrypt(text, text) TO dashboard_user;
3116
3117  --
3118  --
3119  -- Name: FUNCTION pgp_sym_encrypt(text, text, text); Type: ACL; Schema: extensions;
Owner: postgres
3120  --
3121
3122  GRANT ALL ON FUNCTION extensions.pgp_sym_encrypt(text, text, text) TO dashboard_user;
3123
3124  --
3125  --
3126  -- Name: FUNCTION pgp_sym_encrypt_bytea(bytea, text); Type: ACL; Schema: extensions;
Owner: postgres
3127  --
3128
3129  GRANT ALL ON FUNCTION extensions.pgp_sym_encrypt_bytea(bytea, text) TO dashboard_user;
3130
3131  --
3132
```

```
3133  -- Name: FUNCTION pgp_sym_encrypt_bytea(bytea, text, text); Type: ACL; Schema:
3134  extensions; Owner: postgres
3135  --
3136  GRANT ALL ON FUNCTION extensions.pgp_sym_encrypt_bytea(bytea, text, text) TO
3137  dashboard_user;
3138  --
3139  --
3140  -- Name: FUNCTION sign(payload json, secret text, algorithm text); Type: ACL; Schema:
3141  extensions; Owner: postgres
3142  --
3143  GRANT ALL ON FUNCTION extensions.sign(payload json, secret text, algorithm text) TO
3144  dashboard_user;
3145  --
3146  --
3147  -- Name: FUNCTION try_cast_double(inp text); Type: ACL; Schema: extensions; Owner:
3148  postgres
3149  --
3150  GRANT ALL ON FUNCTION extensions.try_cast_double(inp text) TO dashboard_user;
3151  --
3152  --
3153  --
3154  -- Name: FUNCTION url_decode(data text); Type: ACL; Schema: extensions; Owner: postgres
3155  --
3156  GRANT ALL ON FUNCTION extensions.url_decode(data text) TO dashboard_user;
3157  --
3158  --
3159  --
3160  -- Name: FUNCTION url_encode(data bytea); Type: ACL; Schema: extensions; Owner: postgres
3161  --
3162  GRANT ALL ON FUNCTION extensions.url_encode(data bytea) TO dashboard_user;
3163  --
3164  --
3165  --
3166  -- Name: FUNCTION uuid_generate_v1(); Type: ACL; Schema: extensions; Owner: postgres
3167  --
3168  GRANT ALL ON FUNCTION extensions.uuid_generate_v1() TO dashboard_user;
3169  --
3170  --
3171  -- Name: FUNCTION uuid_generate_v1mc(); Type: ACL; Schema: extensions; Owner: postgres
3172  --
3173  GRANT ALL ON FUNCTION extensions.uuid_generate_v1mc() TO dashboard_user;
3174  --
3175  --
3176  -- Name: FUNCTION uuid_generate_v3(namespace uuid, name text); Type: ACL; Schema:
3177  extensions; Owner: postgres
3178  --
3179  GRANT ALL ON FUNCTION extensions.uuid_generate_v3(namespace uuid, name text) TO
3180  dashboard_user;
3181  --
3182  --
3183  -- Name: FUNCTION uuid_generate_v4(); Type: ACL; Schema: extensions; Owner: postgres
3184  --
3185  GRANT ALL ON FUNCTION extensions.uuid_generate_v4(); Type: ACL; Schema: extensions; Owner: postgres
```



```
3190  --
3191
3192  GRANT ALL ON FUNCTION extensions.uuid_generate_v4() TO dashboard_user;
3193
3194  --
3195  -- Name: FUNCTION uuid_generate_v5(namespace uuid, name text); Type: ACL; Schema:
3196  extensions; Owner: postgres
3197  --
3198
3199  GRANT ALL ON FUNCTION extensions.uuid_generate_v5(namespace uuid, name text) TO
3200  dashboard_user;
3201
3202  --
3203  -- Name: FUNCTION uuid_nil(); Type: ACL; Schema: extensions; Owner: postgres
3204  --
3205
3206  GRANT ALL ON FUNCTION extensions.uuid_nil() TO dashboard_user;
3207
3208  --
3209  -- Name: FUNCTION uuid_ns_dns(); Type: ACL; Schema: extensions; Owner: postgres
3210  --
3211
3212  GRANT ALL ON FUNCTION extensions.uuid_ns_dns() TO dashboard_user;
3213
3214  --
3215  -- Name: FUNCTION uuid_ns_oid(); Type: ACL; Schema: extensions; Owner: postgres
3216  --
3217
3218  GRANT ALL ON FUNCTION extensions.uuid_ns_oid() TO dashboard_user;
3219
3220  --
3221  -- Name: FUNCTION uuid_ns_url(); Type: ACL; Schema: extensions; Owner: postgres
3222  --
3223
3224  GRANT ALL ON FUNCTION extensions.uuid_ns_url() TO dashboard_user;
3225
3226  --
3227  -- Name: FUNCTION uuid_ns_x500(); Type: ACL; Schema: extensions; Owner: postgres
3228  --
3229
3230  GRANT ALL ON FUNCTION extensions.uuid_ns_x500() TO dashboard_user;
3231
3232  --
3233  -- Name: FUNCTION verify(token text, secret text, algorithm text); Type: ACL; Schema:
3234  extensions; Owner: postgres
3235  --
3236
3237  GRANT ALL ON FUNCTION extensions.verify(token text, secret text, algorithm text) TO
3238  dashboard_user;
3239
3240  --
3241  -- Name: FUNCTION rebuild_on_ddl(); Type: ACL; Schema: graphql; Owner: supabase_admin
3242  --
3243
3244  GRANT ALL ON FUNCTION graphql.rebuild_on_ddl() TO postgres;
3245  GRANT ALL ON FUNCTION graphql.rebuild_on_ddl() TO anon;
```

```

3250 GRANT ALL ON FUNCTION graphql.rebuild_on_ddl() TO authenticated;
3251 GRANT ALL ON FUNCTION graphql.rebuild_on_ddl() TO service_role;
3252
3253
3254 --
3255 -- Name: FUNCTION rebuild_on_drop(); Type: ACL; Schema: graphql; Owner: supabase_admin
3256 --
3257
3258 GRANT ALL ON FUNCTION graphql.rebuild_on_drop() TO postgres;
3259 GRANT ALL ON FUNCTION graphql.rebuild_on_drop() TO anon;
3260 GRANT ALL ON FUNCTION graphql.rebuild_on_drop() TO authenticated;
3261 GRANT ALL ON FUNCTION graphql.rebuild_on_drop() TO service_role;
3262
3263
3264 --
3265 -- Name: FUNCTION rebuild_schema(); Type: ACL; Schema: graphql; Owner: supabase_admin
3266 --
3267
3268 GRANT ALL ON FUNCTION graphql.rebuild_schema() TO postgres;
3269 GRANT ALL ON FUNCTION graphql.rebuild_schema() TO anon;
3270 GRANT ALL ON FUNCTION graphql.rebuild_schema() TO authenticated;
3271 GRANT ALL ON FUNCTION graphql.rebuild_schema() TO service_role;
3272
3273
3274 --
3275 -- Name: FUNCTION variable_definitions_sort(variable_definitions jsonb); Type: ACL;
Schema: graphql; Owner: supabase_admin
3276 --
3277
3278 GRANT ALL ON FUNCTION graphql.variable_definitions_sort(variable_definitions jsonb) TO
postgres;
3279 GRANT ALL ON FUNCTION graphql.variable_definitions_sort(variable_definitions jsonb) TO
anon;
3280 GRANT ALL ON FUNCTION graphql.variable_definitions_sort(variable_definitions jsonb) TO
authenticated;
3281 GRANT ALL ON FUNCTION graphql.variable_definitions_sort(variable_definitions jsonb) TO
service_role;
3282
3283
3284 --
3285 -- Name: FUNCTION get_auth(p_username text); Type: ACL; Schema: pgbouncer; Owner: postgres
3286 --
3287
3288 REVOKE ALL ON FUNCTION pgbouncer.get_auth(p_username text) FROM PUBLIC;
3289 GRANT ALL ON FUNCTION pgbouncer.get_auth(p_username text) TO pgbouncer;
3290
3291
3292 --
3293 -- Name: FUNCTION apply_ri(wal jsonb, max_record_bytes integer); Type: ACL; Schema:
realtime; Owner: supabase_admin
3294 --
3295
3296 GRANT ALL ON FUNCTION realtime.apply_ri(wal jsonb, max_record_bytes integer) TO
postgres;
3297 GRANT ALL ON FUNCTION realtime.apply_ri(wal jsonb, max_record_bytes integer) TO
dashboard_user;
3298
3299
3300 --
3301 -- Name: FUNCTION build_prepared_statement_sql(prepared_statement_name text, entity
regclass, columns realtime.wal_column[]); Type: ACL; Schema: realtime; Owner:
supabase_admin
3302 --
3303

```

```
3304 GRANT ALL ON FUNCTION realtime.build_prepared_statement_sql(prepared_statement_name text
, entity regclass, columns realtime.wal_column[]) TO postgres;
3305 GRANT ALL ON FUNCTION realtime.build_prepared_statement_sql(prepared_statement_name text
, entity regclass, columns realtime.wal_column[]) TO dashboard_user;
3306
3307
3308 --
3309 -- Name: FUNCTION "cast"(val text, type_ regtype); Type: ACL; Schema: realtime; Owner:
supabase_admin
3310 --
3311
3312 GRANT ALL ON FUNCTION realtime."cast"(val text, type_ regtype) TO postgres;
3313 GRANT ALL ON FUNCTION realtime."cast"(val text, type_ regtype) TO dashboard_user;
3314
3315
3316 --
3317 -- Name: FUNCTION check_equality_op(op realtime.equality_op, type_ regtype, val_1 text,
val_2 text); Type: ACL; Schema: realtime; Owner: supabase_admin
3318 --
3319
3320 GRANT ALL ON FUNCTION realtime.check_equality_op(op realtime.equality_op, type_ regtype,
val_1 text, val_2 text) TO postgres;
3321 GRANT ALL ON FUNCTION realtime.check_equality_op(op realtime.equality_op, type_ regtype,
val_1 text, val_2 text) TO dashboard_user;
3322
3323
3324 --
3325 -- Name: FUNCTION is_visible_through_filters(columns realtime.wal_column[], filters
realtime.user_defined_filter[]); Type: ACL; Schema: realtime; Owner: supabase_admin
3326 --
3327
3328 GRANT ALL ON FUNCTION realtime.is_visible_through_filters(columns realtime.wal_column[],
filters realtime.user_defined_filter[]) TO postgres;
3329 GRANT ALL ON FUNCTION realtime.is_visible_through_filters(columns realtime.wal_column[],
filters realtime.user_defined_filter[]) TO dashboard_user;
3330
3331
3332 --
3333 -- Name: FUNCTION quote_wal2json(entity regclass); Type: ACL; Schema: realtime; Owner:
supabase_admin
3334 --
3335
3336 GRANT ALL ON FUNCTION realtime.quote_wal2json(entity regclass) TO postgres;
3337 GRANT ALL ON FUNCTION realtime.quote_wal2json(entity regclass) TO dashboard_user;
3338
3339
3340 --
3341 -- Name: FUNCTION subscription_check_filters(); Type: ACL; Schema: realtime; Owner:
supabase_admin
3342 --
3343
3344 GRANT ALL ON FUNCTION realtime.subscription_check_filters() TO postgres;
3345 GRANT ALL ON FUNCTION realtime.subscription_check_filters() TO dashboard_user;
3346
3347
3348 --
3349 -- Name: FUNCTION to_regrole(role_name text); Type: ACL; Schema: realtime; Owner:
supabase_admin
3350 --
3351
3352 GRANT ALL ON FUNCTION realtime.to_regrole(role_name text) TO postgres;
3353 GRANT ALL ON FUNCTION realtime.to_regrole(role_name text) TO dashboard_user;
3354
3355
```

```

3356  --
3357  -- Name: FUNCTION extension(name text); Type: ACL; Schema: storage; Owner:
supabase_storage_admin
3358  --
3359
3360  GRANT ALL ON FUNCTION storage.extension(name text) TO anon;
3361  GRANT ALL ON FUNCTION storage.extension(name text) TO authenticated;
3362  GRANT ALL ON FUNCTION storage.extension(name text) TO service_role;
3363  GRANT ALL ON FUNCTION storage.extension(name text) TO dashboard_user;
3364  GRANT ALL ON FUNCTION storage.extension(name text) TO postgres;
3365
3366
3367  --
3368  -- Name: FUNCTION filename(name text); Type: ACL; Schema: storage; Owner:
supabase_storage_admin
3369  --
3370
3371  GRANT ALL ON FUNCTION storage.filename(name text) TO anon;
3372  GRANT ALL ON FUNCTION storage.filename(name text) TO authenticated;
3373  GRANT ALL ON FUNCTION storage.filename(name text) TO service_role;
3374  GRANT ALL ON FUNCTION storage.filename(name text) TO dashboard_user;
3375  GRANT ALL ON FUNCTION storage.filename(name text) TO postgres;
3376
3377
3378  --
3379  -- Name: FUNCTION foldername(name text); Type: ACL; Schema: storage; Owner:
supabase_storage_admin
3380  --
3381
3382  GRANT ALL ON FUNCTION storage.foldername(name text) TO anon;
3383  GRANT ALL ON FUNCTION storage.foldername(name text) TO authenticated;
3384  GRANT ALL ON FUNCTION storage.foldername(name text) TO service_role;
3385  GRANT ALL ON FUNCTION storage.foldername(name text) TO dashboard_user;
3386  GRANT ALL ON FUNCTION storage.foldername(name text) TO postgres;
3387
3388
3389  --
3390  -- Name: TABLE audit_log_entries; Type: ACL; Schema: auth; Owner: supabase_auth_admin
3391  --
3392
3393  GRANT ALL ON TABLE auth.audit_log_entries TO dashboard_user;
3394  GRANT ALL ON TABLE auth.audit_log_entries TO postgres;
3395
3396
3397  --
3398  -- Name: TABLE identities; Type: ACL; Schema: auth; Owner: supabase_auth_admin
3399  --
3400
3401  GRANT ALL ON TABLE auth.identities TO postgres;
3402  GRANT ALL ON TABLE auth.identities TO dashboard_user;
3403
3404
3405  --
3406  -- Name: TABLE instances; Type: ACL; Schema: auth; Owner: supabase_auth_admin
3407  --
3408
3409  GRANT ALL ON TABLE auth.instances TO dashboard_user;
3410  GRANT ALL ON TABLE auth.instances TO postgres;
3411
3412
3413  --
3414  -- Name: TABLE refresh_tokens; Type: ACL; Schema: auth; Owner: supabase_auth_admin
3415  --
3416

```

```
3417 GRANT ALL ON TABLE auth.refresh_tokens TO dashboard_user;
3418 GRANT ALL ON TABLE auth.refresh_tokens TO postgres;
3419
3420
3421 --
3422 -- Name: SEQUENCE refresh_tokens_id_seq; Type: ACL; Schema: auth; Owner:
supabase_auth_admin
3423 --
3424
3425 GRANT ALL ON SEQUENCE auth.refresh_tokens_id_seq TO dashboard_user;
3426 GRANT ALL ON SEQUENCE auth.refresh_tokens_id_seq TO postgres;
3427
3428
3429 --
3430 -- Name: TABLE schema_migrations; Type: ACL; Schema: auth; Owner: supabase_auth_admin
3431 --
3432
3433 GRANT ALL ON TABLE auth.schema_migrations TO dashboard_user;
3434 GRANT ALL ON TABLE auth.schema_migrations TO postgres;
3435
3436
3437 --
3438 -- Name: TABLE users; Type: ACL; Schema: auth; Owner: supabase_auth_admin
3439 --
3440
3441 GRANT ALL ON TABLE auth.users TO dashboard_user;
3442 GRANT ALL ON TABLE auth.users TO postgres;
3443
3444
3445 --
3446 -- Name: TABLE pg_stat_statements; Type: ACL; Schema: extensions; Owner: postgres
3447 --
3448
3449 GRANT ALL ON TABLE extensions.pg_stat_statements TO dashboard_user;
3450
3451
3452 --
3453 -- Name: TABLE pg_stat_statements_info; Type: ACL; Schema: extensions; Owner: postgres
3454 --
3455
3456 GRANT ALL ON TABLE extensions.pg_stat_statements_info TO dashboard_user;
3457
3458
3459 --
3460 -- Name: TABLE "Decisions"; Type: ACL; Schema: public; Owner: supabase_admin
3461 --
3462
3463 GRANT ALL ON TABLE public."Decisions" TO postgres;
3464 GRANT ALL ON TABLE public."Decisions" TO anon;
3465 GRANT ALL ON TABLE public."Decisions" TO authenticated;
3466 GRANT ALL ON TABLE public."Decisions" TO service_role;
3467
3468
3469 --
3470 -- Name: SEQUENCE "Decision_id_seq"; Type: ACL; Schema: public; Owner: supabase_admin
3471 --
3472
3473 GRANT ALL ON SEQUENCE public."Decision_id_seq" TO postgres;
3474 GRANT ALL ON SEQUENCE public."Decision_id_seq" TO anon;
3475 GRANT ALL ON SEQUENCE public."Decision_id_seq" TO authenticated;
3476 GRANT ALL ON SEQUENCE public."Decision_id_seq" TO service_role;
3477
3478
3479 --
```

```
3480 -- Name: TABLE "Donations"; Type: ACL; Schema: public; Owner: supabase_admin
3481 --
3482
3483 GRANT ALL ON TABLE public."Donations" TO postgres;
3484 GRANT ALL ON TABLE public."Donations" TO anon;
3485 GRANT ALL ON TABLE public."Donations" TO authenticated;
3486 GRANT ALL ON TABLE public."Donations" TO service_role;
3487
3488
3489 --
3490 -- Name: SEQUENCE "Donation_id_seq"; Type: ACL; Schema: public; Owner: supabase_admin
3491 --
3492
3493 GRANT ALL ON SEQUENCE public."Donation_id_seq" TO postgres;
3494 GRANT ALL ON SEQUENCE public."Donation_id_seq" TO anon;
3495 GRANT ALL ON SEQUENCE public."Donation_id_seq" TO authenticated;
3496 GRANT ALL ON SEQUENCE public."Donation_id_seq" TO service_role;
3497
3498
3499 --
3500 -- Name: TABLE "Donors"; Type: ACL; Schema: public; Owner: supabase_admin
3501 --
3502
3503 GRANT ALL ON TABLE public."Donors" TO postgres;
3504 GRANT ALL ON TABLE public."Donors" TO anon;
3505 GRANT ALL ON TABLE public."Donors" TO authenticated;
3506 GRANT ALL ON TABLE public."Donors" TO service_role;
3507
3508
3509 --
3510 -- Name: SEQUENCE "Donor_id_seq"; Type: ACL; Schema: public; Owner: supabase_admin
3511 --
3512
3513 GRANT ALL ON SEQUENCE public."Donor_id_seq" TO postgres;
3514 GRANT ALL ON SEQUENCE public."Donor_id_seq" TO anon;
3515 GRANT ALL ON SEQUENCE public."Donor_id_seq" TO authenticated;
3516 GRANT ALL ON SEQUENCE public."Donor_id_seq" TO service_role;
3517
3518
3519 --
3520 -- Name: TABLE "Governors"; Type: ACL; Schema: public; Owner: supabase_admin
3521 --
3522
3523 GRANT ALL ON TABLE public."Governors" TO postgres;
3524 GRANT ALL ON TABLE public."Governors" TO anon;
3525 GRANT ALL ON TABLE public."Governors" TO authenticated;
3526 GRANT ALL ON TABLE public."Governors" TO service_role;
3527
3528
3529 --
3530 -- Name: SEQUENCE "Governors_id_seq"; Type: ACL; Schema: public; Owner: supabase_admin
3531 --
3532
3533 GRANT ALL ON SEQUENCE public."Governors_id_seq" TO postgres;
3534 GRANT ALL ON SEQUENCE public."Governors_id_seq" TO anon;
3535 GRANT ALL ON SEQUENCE public."Governors_id_seq" TO authenticated;
3536 GRANT ALL ON SEQUENCE public."Governors_id_seq" TO service_role;
3537
3538
3539 --
3540 -- Name: TABLE "Organizations"; Type: ACL; Schema: public; Owner: supabase_admin
3541 --
3542
3543 GRANT ALL ON TABLE public."Organizations" TO postgres;
```

```
3544 GRANT ALL ON TABLE public."Organizations" TO anon;
3545 GRANT ALL ON TABLE public."Organizations" TO authenticated;
3546 GRANT ALL ON TABLE public."Organizations" TO service_role;
3547
3548
3549 --
3550 -- Name: SEQUENCE "Organizations_id_seq"; Type: ACL; Schema: public; Owner:
supabase_admin
3551 --
3552
3553 GRANT ALL ON SEQUENCE public."Organizations_id_seq" TO postgres;
3554 GRANT ALL ON SEQUENCE public."Organizations_id_seq" TO anon;
3555 GRANT ALL ON SEQUENCE public."Organizations_id_seq" TO authenticated;
3556 GRANT ALL ON SEQUENCE public."Organizations_id_seq" TO service_role;
3557
3558
3559 --
3560 -- Name: TABLE "Projects"; Type: ACL; Schema: public; Owner: supabase_admin
3561 --
3562
3563 GRANT ALL ON TABLE public."Projects" TO postgres;
3564 GRANT ALL ON TABLE public."Projects" TO anon;
3565 GRANT ALL ON TABLE public."Projects" TO authenticated;
3566 GRANT ALL ON TABLE public."Projects" TO service_role;
3567
3568
3569 --
3570 -- Name: SEQUENCE "Projects_id_seq"; Type: ACL; Schema: public; Owner: supabase_admin
3571 --
3572
3573 GRANT ALL ON SEQUENCE public."Projects_id_seq" TO postgres;
3574 GRANT ALL ON SEQUENCE public."Projects_id_seq" TO anon;
3575 GRANT ALL ON SEQUENCE public."Projects_id_seq" TO authenticated;
3576 GRANT ALL ON SEQUENCE public."Projects_id_seq" TO service_role;
3577
3578
3579 --
3580 -- Name: TABLE "SingleVotes"; Type: ACL; Schema: public; Owner: supabase_admin
3581 --
3582
3583 GRANT ALL ON TABLE public."SingleVotes" TO postgres;
3584 GRANT ALL ON TABLE public."SingleVotes" TO anon;
3585 GRANT ALL ON TABLE public."SingleVotes" TO authenticated;
3586 GRANT ALL ON TABLE public."SingleVotes" TO service_role;
3587
3588
3589 --
3590 -- Name: SEQUENCE "SingleVote_id_seq"; Type: ACL; Schema: public; Owner: supabase_admin
3591 --
3592
3593 GRANT ALL ON SEQUENCE public."SingleVote_id_seq" TO postgres;
3594 GRANT ALL ON SEQUENCE public."SingleVote_id_seq" TO anon;
3595 GRANT ALL ON SEQUENCE public."SingleVote_id_seq" TO authenticated;
3596 GRANT ALL ON SEQUENCE public."SingleVote_id_seq" TO service_role;
3597
3598
3599 --
3600 -- Name: TABLE "Solutions"; Type: ACL; Schema: public; Owner: supabase_admin
3601 --
3602
3603 GRANT ALL ON TABLE public."Solutions" TO postgres;
3604 GRANT ALL ON TABLE public."Solutions" TO anon;
3605 GRANT ALL ON TABLE public."Solutions" TO authenticated;
3606 GRANT ALL ON TABLE public."Solutions" TO service_role;
```

```
3607
3608
3609 --
3610 -- Name: SEQUENCE "Solutions_id_seq"; Type: ACL; Schema: public; Owner: supabase_admin
3611 --
3612
3613 GRANT ALL ON SEQUENCE public."Solutions_id_seq" TO postgres;
3614 GRANT ALL ON SEQUENCE public."Solutions_id_seq" TO anon;
3615 GRANT ALL ON SEQUENCE public."Solutions_id_seq" TO authenticated;
3616 GRANT ALL ON SEQUENCE public."Solutions_id_seq" TO service_role;
3617
3618
3619 --
3620 -- Name: TABLE "SystemConstants"; Type: ACL; Schema: public; Owner: supabase_admin
3621 --
3622
3623 GRANT ALL ON TABLE public."SystemConstants" TO postgres;
3624 GRANT ALL ON TABLE public."SystemConstants" TO anon;
3625 GRANT ALL ON TABLE public."SystemConstants" TO authenticated;
3626 GRANT ALL ON TABLE public."SystemConstants" TO service_role;
3627
3628
3629 --
3630 -- Name: SEQUENCE "SystemConstants_id_seq"; Type: ACL; Schema: public; Owner:
supabase_admin
3631 --
3632
3633 GRANT ALL ON SEQUENCE public."SystemConstants_id_seq" TO postgres;
3634 GRANT ALL ON SEQUENCE public."SystemConstants_id_seq" TO anon;
3635 GRANT ALL ON SEQUENCE public."SystemConstants_id_seq" TO authenticated;
3636 GRANT ALL ON SEQUENCE public."SystemConstants_id_seq" TO service_role;
3637
3638
3639 --
3640 -- Name: TABLE schema_migrations; Type: ACL; Schema: realtime; Owner: supabase_admin
3641 --
3642
3643 GRANT ALL ON TABLE realtime.schema_migrations TO postgres;
3644 GRANT ALL ON TABLE realtime.schema_migrations TO dashboard_user;
3645
3646
3647 --
3648 -- Name: TABLE subscription; Type: ACL; Schema: realtime; Owner: supabase_admin
3649 --
3650
3651 GRANT ALL ON TABLE realtime.subscription TO postgres;
3652 GRANT ALL ON TABLE realtime.subscription TO dashboard_user;
3653
3654
3655 --
3656 -- Name: SEQUENCE subscription_id_seq; Type: ACL; Schema: realtime; Owner: supabase_admin
3657 --
3658
3659 GRANT ALL ON SEQUENCE realtime.subscription_id_seq TO postgres;
3660 GRANT ALL ON SEQUENCE realtime.subscription_id_seq TO dashboard_user;
3661
3662
3663 --
3664 -- Name: TABLE buckets; Type: ACL; Schema: storage; Owner: supabase_storage_admin
3665 --
3666
3667 GRANT ALL ON TABLE storage.buckets TO anon;
3668 GRANT ALL ON TABLE storage.buckets TO authenticated;
3669 GRANT ALL ON TABLE storage.buckets TO service_role;
```



```

3670 GRANT ALL ON TABLE storage.buckets TO postgres;
3671
3672
3673 --
3674 -- Name: TABLE migrations; Type: ACL; Schema: storage; Owner: supabase_storage_admin
3675 --
3676
3677 GRANT ALL ON TABLE storage.migrations TO anon;
3678 GRANT ALL ON TABLE storage.migrations TO authenticated;
3679 GRANT ALL ON TABLE storage.migrations TO service_role;
3680 GRANT ALL ON TABLE storage.migrations TO postgres;
3681
3682
3683 --
3684 -- Name: TABLE objects; Type: ACL; Schema: storage; Owner: supabase_storage_admin
3685 --
3686
3687 GRANT ALL ON TABLE storage.objects TO anon;
3688 GRANT ALL ON TABLE storage.objects TO authenticated;
3689 GRANT ALL ON TABLE storage.objects TO service_role;
3690 GRANT ALL ON TABLE storage.objects TO postgres;
3691
3692
3693 --
3694 -- Name: DEFAULT PRIVILEGES FOR SEQUENCES; Type: DEFAULT ACL; Schema: auth; Owner:
supabase_auth_admin
3695 --
3696
3697 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_auth_admin IN SCHEMA auth GRANT ALL ON
SEQUENCES TO postgres;
3698 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_auth_admin IN SCHEMA auth GRANT ALL ON
SEQUENCES TO dashboard_user;
3699
3700
3701 --
3702 -- Name: DEFAULT PRIVILEGES FOR FUNCTIONS; Type: DEFAULT ACL; Schema: auth; Owner:
supabase_auth_admin
3703 --
3704
3705 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_auth_admin IN SCHEMA auth GRANT ALL ON
FUNCTIONS TO postgres;
3706 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_auth_admin IN SCHEMA auth GRANT ALL ON
FUNCTIONS TO dashboard_user;
3707
3708
3709 --
3710 -- Name: DEFAULT PRIVILEGES FOR TABLES; Type: DEFAULT ACL; Schema: auth; Owner:
supabase_auth_admin
3711 --
3712
3713 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_auth_admin IN SCHEMA auth GRANT ALL ON TABLES
TO postgres;
3714 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_auth_admin IN SCHEMA auth GRANT ALL ON TABLES
TO dashboard_user;
3715
3716
3717 --
3718 -- Name: DEFAULT PRIVILEGES FOR SEQUENCES; Type: DEFAULT ACL; Schema: graphql; Owner:
supabase_admin
3719 --
3720
3721 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql GRANT ALL ON
SEQUENCES TO postgres;
3722 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql GRANT ALL ON

```

```

SEQUENCES TO anon;
3723 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql GRANT ALL ON
SEQUENCES TO authenticated;
3724 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql GRANT ALL ON
SEQUENCES TO service_role;

3725
3726
3727 --
3728 -- Name: DEFAULT PRIVILEGES FOR FUNCTIONS; Type: DEFAULT ACL; Schema: graphql; Owner:
supabase_admin
3729 --
3730
3731 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql GRANT ALL ON
FUNCTIONS TO postgres;
3732 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql GRANT ALL ON
FUNCTIONS TO anon;
3733 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql GRANT ALL ON
FUNCTIONS TO authenticated;
3734 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql GRANT ALL ON
FUNCTIONS TO service_role;

3735
3736
3737 --
3738 -- Name: DEFAULT PRIVILEGES FOR TABLES; Type: DEFAULT ACL; Schema: graphql; Owner:
supabase_admin
3739 --
3740
3741 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql GRANT ALL ON TABLES
TO postgres;
3742 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql GRANT ALL ON TABLES
TO anon;
3743 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql GRANT ALL ON TABLES
TO authenticated;
3744 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql GRANT ALL ON TABLES
TO service_role;

3745
3746
3747 --
3748 -- Name: DEFAULT PRIVILEGES FOR SEQUENCES; Type: DEFAULT ACL; Schema: graphql_public;
Owner: supabase_admin
3749 --
3750
3751 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql_public GRANT ALL ON
SEQUENCES TO postgres;
3752 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql_public GRANT ALL ON
SEQUENCES TO anon;
3753 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql_public GRANT ALL ON
SEQUENCES TO authenticated;
3754 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql_public GRANT ALL ON
SEQUENCES TO service_role;

3755
3756
3757 --
3758 -- Name: DEFAULT PRIVILEGES FOR FUNCTIONS; Type: DEFAULT ACL; Schema: graphql_public;
Owner: supabase_admin
3759 --
3760
3761 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql_public GRANT ALL ON
FUNCTIONS TO postgres;
3762 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql_public GRANT ALL ON
FUNCTIONS TO anon;
3763 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql_public GRANT ALL ON
FUNCTIONS TO authenticated;
3764 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql_public GRANT ALL ON

```

```

FUNCTIONS  TO service_role;
3765
3766
3767 --
3768 -- Name: DEFAULT PRIVILEGES FOR TABLES; Type: DEFAULT ACL; Schema: graphql_public;
Owner: supabase_admin
3769 --
3770
3771 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql_public GRANT ALL ON
TABLES  TO postgres;
3772 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql_public GRANT ALL ON
TABLES  TO anon;
3773 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql_public GRANT ALL ON
TABLES  TO authenticated;
3774 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA graphql_public GRANT ALL ON
TABLES  TO service_role;
3775
3776
3777 --
3778 -- Name: DEFAULT PRIVILEGES FOR SEQUENCES; Type: DEFAULT ACL; Schema: public; Owner:
postgres
3779 --
3780
3781 ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA public GRANT ALL ON SEQUENCES  TO
postgres;
3782 ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA public GRANT ALL ON SEQUENCES  TO
anon;
3783 ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA public GRANT ALL ON SEQUENCES  TO
authenticated;
3784 ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA public GRANT ALL ON SEQUENCES  TO
service_role;
3785
3786
3787 --
3788 -- Name: DEFAULT PRIVILEGES FOR SEQUENCES; Type: DEFAULT ACL; Schema: public; Owner:
supabase_admin
3789 --
3790
3791 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA public GRANT ALL ON SEQUENCES
TO postgres;
3792 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA public GRANT ALL ON SEQUENCES
TO anon;
3793 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA public GRANT ALL ON SEQUENCES
TO authenticated;
3794 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA public GRANT ALL ON SEQUENCES
TO service_role;
3795
3796
3797 --
3798 -- Name: DEFAULT PRIVILEGES FOR FUNCTIONS; Type: DEFAULT ACL; Schema: public; Owner:
postgres
3799 --
3800
3801 ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA public GRANT ALL ON FUNCTIONS  TO
postgres;
3802 ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA public GRANT ALL ON FUNCTIONS  TO
anon;
3803 ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA public GRANT ALL ON FUNCTIONS  TO
authenticated;
3804 ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA public GRANT ALL ON FUNCTIONS  TO
service_role;
3805
3806
3807 --

```

```

3808  -- Name: DEFAULT PRIVILEGES FOR FUNCTIONS; Type: DEFAULT ACL; Schema: public; Owner:
supabase_admin
3809  --
3810
3811  ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA public GRANT ALL ON FUNCTIONS
    TO postgres;
3812  ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA public GRANT ALL ON FUNCTIONS
    TO anon;
3813  ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA public GRANT ALL ON FUNCTIONS
    TO authenticated;
3814  ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA public GRANT ALL ON FUNCTIONS
    TO service_role;
3815
3816
3817  --
3818  -- Name: DEFAULT PRIVILEGES FOR TABLES; Type: DEFAULT ACL; Schema: public; Owner:
postgres
3819  --
3820
3821  ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA public GRANT ALL ON TABLES TO
postgres;
3822  ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA public GRANT ALL ON TABLES TO anon;
3823  ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA public GRANT ALL ON TABLES TO
authenticated;
3824  ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA public GRANT ALL ON TABLES TO
service_role;
3825
3826
3827  --
3828  -- Name: DEFAULT PRIVILEGES FOR TABLES; Type: DEFAULT ACL; Schema: public; Owner:
supabase_admin
3829  --
3830
3831  ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA public GRANT ALL ON TABLES
    TO postgres;
3832  ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA public GRANT ALL ON TABLES
    TO anon;
3833  ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA public GRANT ALL ON TABLES
    TO authenticated;
3834  ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA public GRANT ALL ON TABLES
    TO service_role;
3835
3836
3837  --
3838  -- Name: DEFAULT PRIVILEGES FOR SEQUENCES; Type: DEFAULT ACL; Schema: realtime; Owner:
supabase_admin
3839  --
3840
3841  ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA realtime GRANT ALL ON
SEQUENCES TO postgres;
3842  ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA realtime GRANT ALL ON
SEQUENCES TO dashboard_user;
3843
3844
3845  --
3846  -- Name: DEFAULT PRIVILEGES FOR FUNCTIONS; Type: DEFAULT ACL; Schema: realtime; Owner:
supabase_admin
3847  --
3848
3849  ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA realtime GRANT ALL ON
FUNCTIONS TO postgres;
3850  ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA realtime GRANT ALL ON
FUNCTIONS TO dashboard_user;
3851

```

```

3852
3853 --
3854 -- Name: DEFAULT PRIVILEGES FOR TABLES; Type: DEFAULT ACL; Schema: realtime; Owner:
supabase_admin
3855 --
3856
3857 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA realtime GRANT ALL ON TABLES
TO postgres;
3858 ALTER DEFAULT PRIVILEGES FOR ROLE supabase_admin IN SCHEMA realtime GRANT ALL ON TABLES
TO dashboard_user;
3859
3860
3861 --
3862 -- Name: DEFAULT PRIVILEGES FOR SEQUENCES; Type: DEFAULT ACL; Schema: storage; Owner:
postgres
3863 --
3864
3865 ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA storage GRANT ALL ON SEQUENCES TO
postgres;
3866 ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA storage GRANT ALL ON SEQUENCES TO
anon;
3867 ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA storage GRANT ALL ON SEQUENCES TO
authenticated;
3868 ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA storage GRANT ALL ON SEQUENCES TO
service_role;
3869
3870
3871 --
3872 -- Name: DEFAULT PRIVILEGES FOR FUNCTIONS; Type: DEFAULT ACL; Schema: storage; Owner:
postgres
3873 --
3874
3875 ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA storage GRANT ALL ON FUNCTIONS TO
postgres;
3876 ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA storage GRANT ALL ON FUNCTIONS TO
anon;
3877 ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA storage GRANT ALL ON FUNCTIONS TO
authenticated;
3878 ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA storage GRANT ALL ON FUNCTIONS TO
service_role;
3879
3880
3881 --
3882 -- Name: DEFAULT PRIVILEGES FOR TABLES; Type: DEFAULT ACL; Schema: storage; Owner:
postgres
3883 --
3884
3885 ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA storage GRANT ALL ON TABLES TO
postgres;
3886 ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA storage GRANT ALL ON TABLES TO
anon;
3887 ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA storage GRANT ALL ON TABLES TO
authenticated;
3888 ALTER DEFAULT PRIVILEGES FOR ROLE postgres IN SCHEMA storage GRANT ALL ON TABLES TO
service_role;
3889
3890
3891 --
3892 -- Name: issue_graphql_placeholder; Type: EVENT TRIGGER; Schema: -; Owner: supabase_admin
3893 --
3894
3895 CREATE EVENT TRIGGER issue_graphql_placeholder ON sql_drop
3896 WHEN TAG IN ('DROP EXTENSION')
3897 EXECUTE FUNCTION extensions.set_graphql_placeholder();

```

```
3898
3899
3900 ALTER EVENT TRIGGER issue_graphql_placeholder OWNER TO supabase_admin;
3901
3902 --
3903 -- Name: issue_pg_cron_access; Type: EVENT TRIGGER; Schema: -; Owner: postgres
3904 --
3905
3906 CREATE EVENT TRIGGER issue_pg_cron_access ON ddl_command_end
3907     WHEN TAG IN ('CREATE SCHEMA')
3908     EXECUTE FUNCTION extensions.grant_pg_cron_access();
3909
3910
3911 ALTER EVENT TRIGGER issue_pg_cron_access OWNER TO postgres;
3912
3913 --
3914 -- Name: issue_pg_graphql_access; Type: EVENT TRIGGER; Schema: -; Owner: supabase_admin
3915 --
3916
3917 CREATE EVENT TRIGGER issue_pg_graphql_access ON ddl_command_end
3918     WHEN TAG IN ('CREATE FUNCTION')
3919     EXECUTE FUNCTION extensions.grant_pg_graphql_access();
3920
3921
3922 ALTER EVENT TRIGGER issue_pg_graphql_access OWNER TO supabase_admin;
3923
3924 --
3925 -- Name: issue_pg_net_access; Type: EVENT TRIGGER; Schema: -; Owner: postgres
3926 --
3927
3928 CREATE EVENT TRIGGER issue_pg_net_access ON ddl_command_end
3929     WHEN TAG IN ('CREATE EXTENSION')
3930     EXECUTE FUNCTION extensions.grant_pg_net_access();
3931
3932
3933 ALTER EVENT TRIGGER issue_pg_net_access OWNER TO postgres;
3934
3935 --
3936 -- Name: pgrst_ddl_watch; Type: EVENT TRIGGER; Schema: -; Owner: supabase_admin
3937 --
3938
3939 CREATE EVENT TRIGGER pgrst_ddl_watch ON ddl_command_end
3940     EXECUTE FUNCTION extensions.pgrst_ddl_watch();
3941
3942
3943 ALTER EVENT TRIGGER pgrst_ddl_watch OWNER TO supabase_admin;
3944
3945 --
3946 -- Name: pgrst_drop_watch; Type: EVENT TRIGGER; Schema: -; Owner: supabase_admin
3947 --
3948
3949 CREATE EVENT TRIGGER pgrst_drop_watch ON sql_drop
3950     EXECUTE FUNCTION extensions.pgrst_drop_watch();
3951
3952
3953 ALTER EVENT TRIGGER pgrst_drop_watch OWNER TO supabase_admin;
3954
3955 --
3956 -- PostgreSQL database dump complete
3957 --
3958
3959
```