Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Μ Α Θ Η Μ Α
ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

Ε Ρ Γ Α Σ Ι Α
ΕΞΑΜΗΝΙΑΙΑ ΕΡΓΑΣΙΑ (ΠΑΡΑΔΟΤΕΟ 2)

Σ Π Ο Υ Δ Α Σ Τ Ε Σ
ΚΑΡΑΜ ΚΩΝΣΤΑΝΤΙΝΟΣ
ΚΟΛΙΟΣ ΑΠΟΣΤΟΛΟΣ

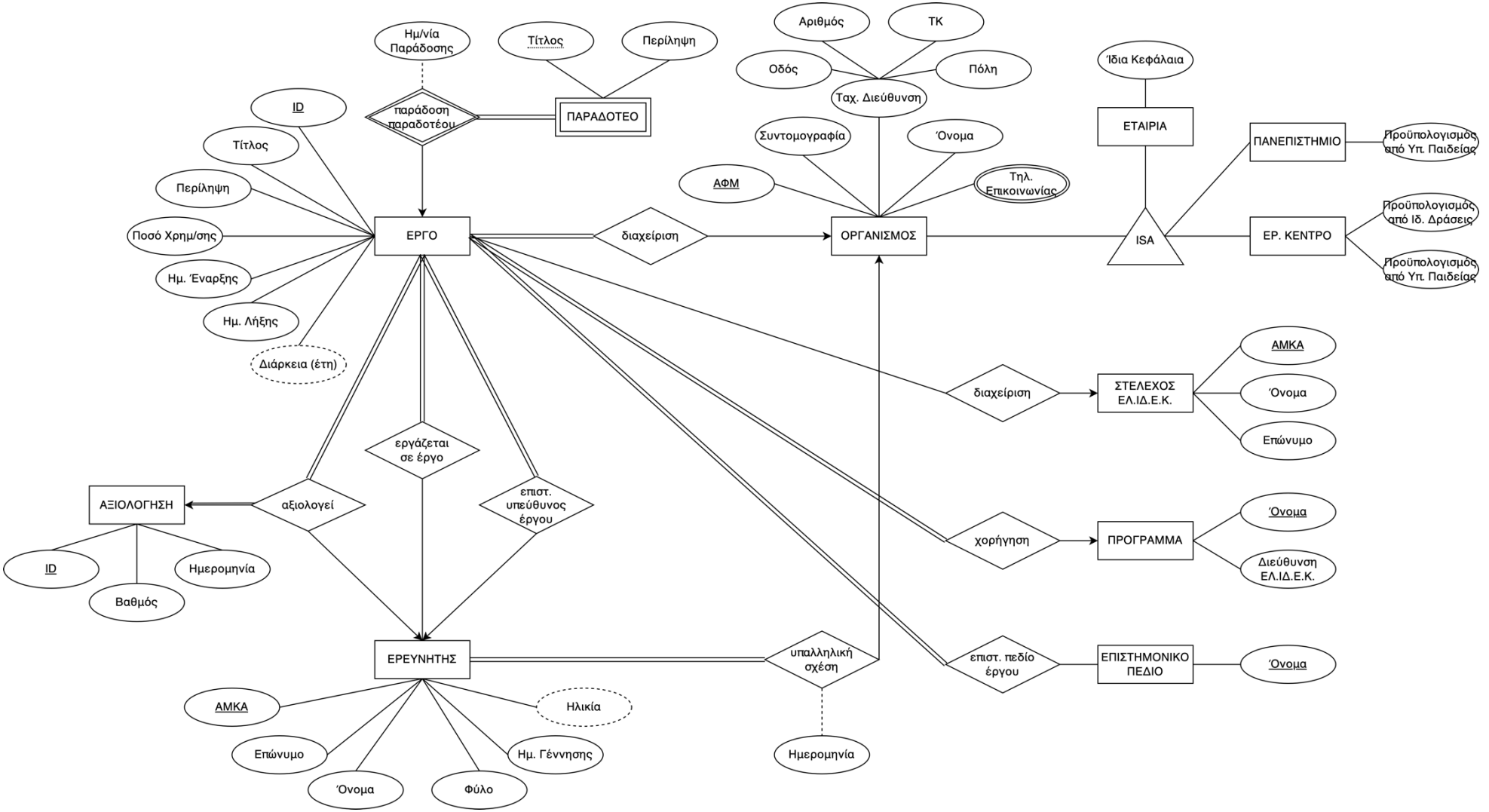
Σ Υ Ν Δ Ε Σ Μ Ο Σ Ε Φ Α Ρ Μ Ο Γ Η Σ
https://dblab2022.gtsb.io/

2 0 2 1 – 2 2

Περιεχόμενα:

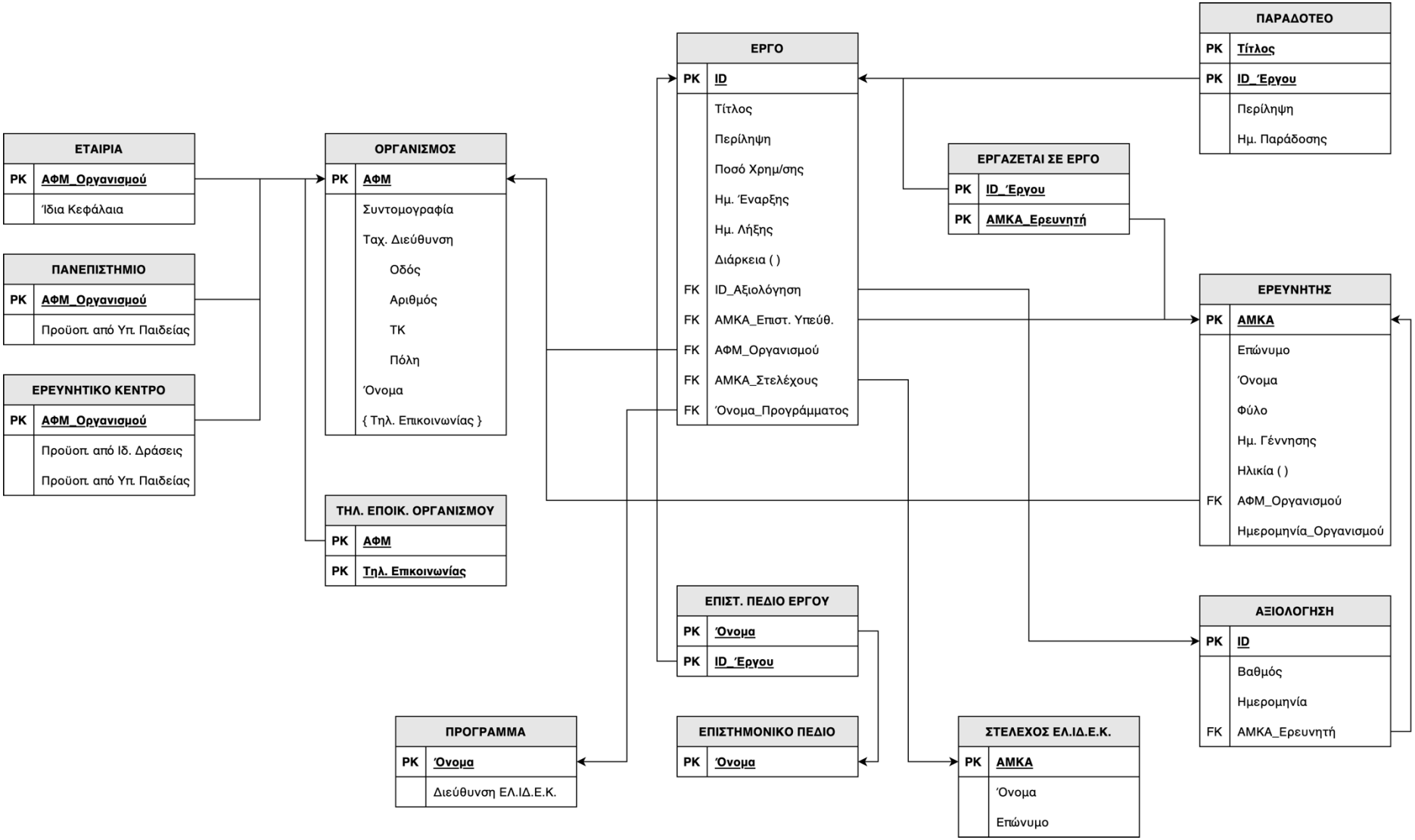
1.1 ER Διάγραμμα

Το νέο ER Διάγραμμα που χρησιμοποιήθηκε έπειτα από τροποποιήσεις:



1.2 Relational Διάγραμμα

Το Σχεσιακό Διάγραμμα όπως προέκυψε από το παραπάνω ER:



- Ο πίνακας «Εργάζεται σε Έργο» είναι μία σχέση πολλά προς πολλά (πολλοί ερευνητές εργάζονται σε πολλά έργα) που συσχετίζει τις οντότητες Έργο και Ερευνητής. Έτσι δημιουργήσαμε μία νέα οντότητα η οποία περιέχει ως primary key δύο foreign keys, ένα προς το έργο (ID) και ένα προς τον ερευνητή (ΑΜΚΑ).
- Για τον ίδιο λόγο δημιουργήσαμε και τον πίνακα «Επιστημονικό Πεδίο Έργου» που συσχετίζει τις οντότητες Έργο και Επιστημονικό Πεδίο, καθώς ένα έργο θα μπορούσε να ανήκει σε περισσότερα του ενός επιστημονικά πεδία και ένα επιστημονικό πεδίο σε πολλά έργα (σχέση πολλά προς πολλά). Έτσι δημιουργήσαμε την οντότητα επιστημονικό πεδίο έργου η οποία έχει ως primary keys δύο foreign keys, ένα προς το έργο (ID) και ένα προς το επιστημονικό πεδίο (Όνομα).
- Η σχέση ISA, αντιμετωπίστηκε με την προθήκη ως primary key το ΑΦΜ του Οργανισμού σε κάθε μία από τις τρεις κατηγορίες, Εταιρεία, Πανεπιστήμιο και Ερευνητικό Κέντρο.
- Το attribute Τηλέφωνο Επικοινωνίας ενός Οργανισμού μπορεί να πάρει πολλές τιμές καθώς ένας οργανισμός μπορεί να έχει περισσότερα από ένα τηλέφωνα επικοινωνίας. Έτσι, δημιουργήσαμε την οντότητα «Τηλ. Επικοινωνίας Οργανισμού» η οποία περιέχει ως primary key, το primary key του οργανισμού (ΑΦΜ) και το τηλέφωνο επικοινωνίας του.
- Η οντότητα «Παραδοτέο» είναι weak entity και άρα θα προσθέσουμε ως foreign key όλα τα primary keys της οντότητας που συνδέεται, δηλαδή το ID του Έργου.

2.1 Indexes (ευρετήρια)

Τα indexes που χρησιμοποιήθηκαν για την γρηγορότερη προσπέλαση των δεδομένων είναι τα εξής:

```
CREATE TABLE thlefwno(
   orgid         CHAR(11)    NOT NULL,
   number        VARCHAR(20) NOT NULL,
   PRIMARY KEY(orgid, number),
   CONSTRAINT fk_org FOREIGN KEY(orgid) REFERENCES organismos(id)
);
-- για την γρήγορη εύρεση των τηλεφώνων ενός οργανισμού
CREATE INDEX idx_thlefwno_id ON thlefwno(orgid);


CREATE TABLE ergazetai_se_ergo(
   ergoid INT  NOT NULL,
   erevnitisid CHAR(11)  NOT NULL,
   UNIQUE (ergoid, erevnitisid),
   CONSTRAINT fk_ergo FOREIGN KEY(ergoid) REFERENCES ergo(id),
   CONSTRAINT fk_erevnitis FOREIGN KEY(erevnitisid) REFERENCES erevnitis(id)
);
CREATE INDEX idx_ergazetai_se_ergo_ergoid ON ergazetai_se_ergo(ergoid);
CREATE INDEX idx_ergazetai_se_ergo_erevnitisid ON ergazetai_se_ergo(erevnitisid);
```

2.2 DDL

Αφού φτιάξαμε ένα τοπικό αντίγραφο της βάσης με την εντολή:

sudo -u postgres psql postgres <create_tables.sql

Το DDL δημιουργήθηκε με χρήση της εντολής:

sudo -u postgres pg_dump –schema-only

και αποθηκεύτηκε στο GitLab για μελλοντική αναφορά.

```
--
-- PostgreSQL database dump
--

-- Dumped from database version 9.6.24
-- Dumped by pg_dump version 9.6.24

SET statement_timeout = 0;
SET lock_timeout = 0;
SET idle_in_transaction_session_timeout = 0;
SET client_encoding = 'UTF8';
SET standard_conforming_strings = on;
SELECT pg_catalog.set_config('search_path', '', false);
SET check_function_bodies = false;
SET xmloption = content;
SET client_min_messages = warning;
SET row_security = off;

--
-- Name: DATABASE postgres; Type: COMMENT; Schema: -; Owner: postgres
--

COMMENT ON DATABASE postgres IS 'default administrative connection database';


--
-- Name: plpgsql; Type: EXTENSION; Schema: -; Owner:
--

CREATE EXTENSION IF NOT EXISTS plpgsql WITH SCHEMA pg_catalog;


--
-- Name: EXTENSION plpgsql; Type: COMMENT; Schema: -; Owner:
--

COMMENT ON EXTENSION plpgsql IS 'PL/pgSQL procedural language';
```

```
--
-- Name: erevnitis_hlikia(character); Type: FUNCTION; Schema: public; Owner:
postgres
--

CREATE FUNCTION public.erevnitis_hlikia(erevnitis_id character) RETURNS integer
    LANGUAGE plpgsql
    AS $$
DECLARE
  h INT;
BEGIN
  SELECT
    (EXTRACT (YEAR FROM NOW()) - EXTRACT (YEAR FROM erevnitis.gennisi)) INTO h
   FROM erevnitis WHERE (erevnitis.id = erevnitis_id);
  RETURN h;
END;
$$;


ALTER FUNCTION public.erevnitis_hlikia(erevnitis_id character) OWNER TO postgres;

--
-- Name: ergo_diarkia(integer); Type: FUNCTION; Schema: public; Owner: postgres
--

CREATE FUNCTION public.ergo_diarkia(ergo_id integer) RETURNS integer
    LANGUAGE plpgsql
    AS $$
DECLARE
  d INT;
BEGIN
  SELECT
    (EXTRACT (YEAR FROM ergo.lixi()) - EXTRACT (YEAR FROM ergo.enarxi)) INTO d
   FROM ergo WHERE (ergo.id = ergo_id);
  RETURN d;
END;
$$;


ALTER FUNCTION public.ergo_diarkia(ergo_id integer) OWNER TO postgres;

SET default_tablespace = '';

SET default_with_oids = false;

--
-- Name: axiologiseis; Type: TABLE; Schema: public; Owner: postgres
--
```

```
CREATE TABLE public.axiologiseis (
    axiologisiid integer NOT NULL,
    erevnitisid character(11) NOT NULL,
    ergoid integer NOT NULL,
    axiologitisorgid character(9) NOT NULL,
    ergoorgid character(9) NOT NULL,
    CONSTRAINT ck_axiologitis_ergo CHECK ((axiologitisorgid <> ergoorgid))
);


ALTER TABLE public.axiologiseis OWNER TO postgres;

--
-- Name: axiologisi; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.axiologisi (
    id integer NOT NULL,
    imerominia date NOT NULL,
    vathmos real NOT NULL
);


ALTER TABLE public.axiologisi OWNER TO postgres;

--
-- Name: epist_pedio; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.epist_pedio (
    onoma character varying(100) NOT NULL
);


ALTER TABLE public.epist_pedio OWNER TO postgres;

--
-- Name: epist_pedio_ergou; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.epist_pedio_ergou (
    ergoid integer NOT NULL,
    epist_pedio_onoma character varying(100) NOT NULL
);


ALTER TABLE public.epist_pedio_ergou OWNER TO postgres;

--
```

```
-- Name: erevnitis; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.erevnitis (
    id character(11) NOT NULL,
    eponimo character varying(100) NOT NULL,
    onoma character varying(100) NOT NULL,
    gennisi date NOT NULL,
    fylo character(1) NOT NULL,
    orgid character(9) NOT NULL,
    imerominia_org date NOT NULL,
    CONSTRAINT ck_fylo CHECK (((fylo = 'M'::bpchar) OR (fylo = 'F'::bpchar)))
);


ALTER TABLE public.erevnitis OWNER TO postgres;

--
-- Name: ergazetai_se_ergo; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.ergazetai_se_ergo (
    ergoid integer NOT NULL,
    erevnitisid character(11) NOT NULL
);


ALTER TABLE public.ergazetai_se_ergo OWNER TO postgres;

--
-- Name: erga_ana_erevniti; Type: VIEW; Schema: public; Owner: postgres
--

CREATE VIEW public.erga_ana_erevniti AS
 SELECT ergazetai_se_ergo.ergoid,
    ergazetai_se_ergo.erevnitisid,
    concat(erevnitis.eponimo, ' ', erevnitis.onoma) AS onomateponimo
   FROM (public.ergazetai_se_ergo
     JOIN public.erevnitis ON ((ergazetai_se_ergo.erevnitisid = erevnitis.id)))
  ORDER BY ROW(erevnitis.eponimo, erevnitis.onoma);


ALTER TABLE public.erga_ana_erevniti OWNER TO postgres;

--
-- Name: ergo; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.ergo (
    id integer NOT NULL,
```

```
    poso real NOT NULL,
    titlos character varying(600) NOT NULL,
    perilipsi character varying(1000) NOT NULL,
    enarxi date NOT NULL,
    lixi date NOT NULL,
    stelexosid character(11) NOT NULL,
    programmaonoma character varying(500) NOT NULL,
    epist_ypeyth_ergou character(11) NOT NULL,
    orgid character(9) NOT NULL,
    CONSTRAINT ck_dates1 CHECK ((lixi >= enarxi)),
    CONSTRAINT ck_dates2 CHECK ((((date_part('year'::text, lixi) -
date_part('year'::text, enarxi)) >= (1)::double precision) AND
((date_part('year'::text, lixi) - date_part('year'::text, enarxi)) <= (4)::double
precision)))
);


ALTER TABLE public.ergo OWNER TO postgres;

--
-- Name: hlikia_erevnitwn; Type: VIEW; Schema: public; Owner: postgres
--

CREATE VIEW public.hlikia_erevnitwn AS
 SELECT public.erevnitis_hlikia(erevnitis.id) AS age
   FROM public.erevnitis;


ALTER TABLE public.hlikia_erevnitwn OWNER TO postgres;

--
-- Name: org_erevnitiko_kentro; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.org_erevnitiko_kentro (
    id character(9) NOT NULL,
    proyp_yp_paideias real NOT NULL,
    proyp_idiotikes_draseis real NOT NULL
);


ALTER TABLE public.org_erevnitiko_kentro OWNER TO postgres;

--
-- Name: org_etairia; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.org_etairia (
    id character(9) NOT NULL,
    proyp_idia_kefalaia real NOT NULL
```

```
);


ALTER TABLE public.org_etairia OWNER TO postgres;

--
-- Name: org_panepistimio; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.org_panepistimio (
    id character(9) NOT NULL,
    proyp_yp_paideias real NOT NULL
);


ALTER TABLE public.org_panepistimio OWNER TO postgres;

--
-- Name: organismos; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.organismos (
    id character(11) NOT NULL,
    sintomografia character varying(20) NOT NULL,
    onoma character varying(100) NOT NULL,
    odos character varying(100) NOT NULL,
    poli character varying(50) NOT NULL,
    tk character(5) NOT NULL
);


ALTER TABLE public.organismos OWNER TO postgres;

--
-- Name: paradoteo; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.paradoteo (
    titlos character varying(100) NOT NULL,
    perilipsi character varying(1000) NOT NULL,
    ergoid integer,
    imerominia date NOT NULL
);


ALTER TABLE public.paradoteo OWNER TO postgres;

--
-- Name: programma; Type: TABLE; Schema: public; Owner: postgres
--
```

```
CREATE TABLE public.programma (
    onoma character varying(500) NOT NULL,
    diefthinsi character varying(50) NOT NULL
);


ALTER TABLE public.programma OWNER TO postgres;

--
-- Name: stelexos; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.stelexos (
    id character(11) NOT NULL,
    eponimo character varying(100) NOT NULL,
    onoma character varying(100) NOT NULL
);


ALTER TABLE public.stelexos OWNER TO postgres;

--
-- Name: thlefwno; Type: TABLE; Schema: public; Owner: postgres
--

CREATE TABLE public.thlefwno (
    orgid character(11) NOT NULL,
    number character varying(20) NOT NULL
);


ALTER TABLE public.thlefwno OWNER TO postgres;

--
-- Name: axiologisi axiologisi_pkey; Type: CONSTRAINT; Schema: public; Owner:
postgres
--

ALTER TABLE ONLY public.axiologisi
    ADD CONSTRAINT axiologisi_pkey PRIMARY KEY (id);


--
-- Name: epist_pedio epist_pedio_pkey; Type: CONSTRAINT; Schema: public; Owner:
postgres
--

ALTER TABLE ONLY public.epist_pedio
    ADD CONSTRAINT epist_pedio_pkey PRIMARY KEY (onoma);
```

```
--
-- Name: erevnitis erevnitis_pkey; Type: CONSTRAINT; Schema: public; Owner:
postgres
--

ALTER TABLE ONLY public.erevnitis
    ADD CONSTRAINT erevnitis_pkey PRIMARY KEY (id);


--
-- Name: ergazetai_se_ergo ergazetai_se_ergo_ergoid_erevnitisid_key; Type:
CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.ergazetai_se_ergo
    ADD CONSTRAINT ergazetai_se_ergo_ergoid_erevnitisid_key UNIQUE (ergoid,
erevnitisid);


--
-- Name: ergo ergo_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.ergo
    ADD CONSTRAINT ergo_pkey PRIMARY KEY (id);


--
-- Name: org_erevnitiko_kentro org_erevnitiko_kentro_pkey; Type: CONSTRAINT;
Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.org_erevnitiko_kentro
    ADD CONSTRAINT org_erevnitiko_kentro_pkey PRIMARY KEY (id);


--
-- Name: org_etairia org_etairia_pkey; Type: CONSTRAINT; Schema: public; Owner:
postgres
--

ALTER TABLE ONLY public.org_etairia
    ADD CONSTRAINT org_etairia_pkey PRIMARY KEY (id);


--
-- Name: org_panepistimio org_panepistimio_pkey; Type: CONSTRAINT; Schema: public;
Owner: postgres
--
```

```
ALTER TABLE ONLY public.org_panepistimio
    ADD CONSTRAINT org_panepistimio_pkey PRIMARY KEY (id);


--
-- Name: organismos organismos_pkey; Type: CONSTRAINT; Schema: public; Owner:
postgres
--

ALTER TABLE ONLY public.organismos
    ADD CONSTRAINT organismos_pkey PRIMARY KEY (id);


--
-- Name: paradoteo paradoteo_pkey; Type: CONSTRAINT; Schema: public; Owner:
postgres
--

ALTER TABLE ONLY public.paradoteo
    ADD CONSTRAINT paradoteo_pkey PRIMARY KEY (titlos);


--
-- Name: programma programma_pkey; Type: CONSTRAINT; Schema: public; Owner:
postgres
--

ALTER TABLE ONLY public.programma
    ADD CONSTRAINT programma_pkey PRIMARY KEY (onoma);


--
-- Name: stelexos stelexos_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.stelexos
    ADD CONSTRAINT stelexos_pkey PRIMARY KEY (id);


--
-- Name: thlefwno thlefwno_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.thlefwno
    ADD CONSTRAINT thlefwno_pkey PRIMARY KEY (orgid, number);


--
-- Name: stelexos unq; Type: CONSTRAINT; Schema: public; Owner: postgres
--
```

```
ALTER TABLE ONLY public.stelexos
    ADD CONSTRAINT unq UNIQUE (id, onoma);


--
-- Name: programma unq_diefthinsi; Type: CONSTRAINT; Schema: public; Owner:
postgres
--

ALTER TABLE ONLY public.programma
    ADD CONSTRAINT unq_diefthinsi UNIQUE (onoma, diefthinsi);


--
-- Name: epist_pedio_ergou unq_epist_pedio_ergou_ergoid_epist_pedio_onoma; Type:
CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.epist_pedio_ergou
    ADD CONSTRAINT unq_epist_pedio_ergou_ergoid_epist_pedio_onoma UNIQUE (ergoid,
epist_pedio_onoma);


--
-- Name: axiologiseis unq_erevnitisid; Type: CONSTRAINT; Schema: public; Owner:
postgres
--

ALTER TABLE ONLY public.axiologiseis
    ADD CONSTRAINT unq_erevnitisid UNIQUE (ergoid, erevnitisid);


--
-- Name: ergo unq_orgid; Type: CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.ergo
    ADD CONSTRAINT unq_orgid UNIQUE (id, orgid);


--
-- Name: paradoteo unq_paradoteo; Type: CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.paradoteo
    ADD CONSTRAINT unq_paradoteo UNIQUE (titlos, ergoid);


--
-- Name: ergo unq_programmaonoma; Type: CONSTRAINT; Schema: public; Owner: postgres
```

```
--

ALTER TABLE ONLY public.ergo
    ADD CONSTRAINT unq_programmaonoma UNIQUE (id, programmaonoma);


--
-- Name: idx_ergazetai_se_ergo_erevnitisid; Type: INDEX; Schema: public; Owner:
postgres
--

CREATE INDEX idx_ergazetai_se_ergo_erevnitisid ON public.ergazetai_se_ergo USING
btree (erevnitisid);


--
-- Name: idx_ergazetai_se_ergo_ergoid; Type: INDEX; Schema: public; Owner: postgres
--

CREATE INDEX idx_ergazetai_se_ergo_ergoid ON public.ergazetai_se_ergo USING btree
(ergoid);


--
-- Name: idx_thlefwno_id; Type: INDEX; Schema: public; Owner: postgres
--

CREATE INDEX idx_thlefwno_id ON public.thlefwno USING btree (orgid);


--
-- Name: ergo ergo_epist_ypeyth_ergou_fkey; Type: FK CONSTRAINT; Schema: public;
Owner: postgres
--

ALTER TABLE ONLY public.ergo
    ADD CONSTRAINT ergo_epist_ypeyth_ergou_fkey FOREIGN KEY (epist_ypeyth_ergou)
REFERENCES public.erevnitis(id);


--
-- Name: ergo ergo_epist_ypeyth_ergou_fkey1; Type: FK CONSTRAINT; Schema: public;
Owner: postgres
--

ALTER TABLE ONLY public.ergo
    ADD CONSTRAINT ergo_epist_ypeyth_ergou_fkey1 FOREIGN KEY (epist_ypeyth_ergou)
REFERENCES public.erevnitis(id);


--
```

```
-- Name: ergo ergo_epist_ypeyth_ergou_fkey2; Type: FK CONSTRAINT; Schema: public;
Owner: postgres
--

ALTER TABLE ONLY public.ergo
    ADD CONSTRAINT ergo_epist_ypeyth_ergou_fkey2 FOREIGN KEY (epist_ypeyth_ergou)
REFERENCES public.erevnitis(id);


--
-- Name: ergo ergo_epist_ypeyth_ergou_fkey3; Type: FK CONSTRAINT; Schema: public;
Owner: postgres
--

ALTER TABLE ONLY public.ergo
    ADD CONSTRAINT ergo_epist_ypeyth_ergou_fkey3 FOREIGN KEY (epist_ypeyth_ergou)
REFERENCES public.erevnitis(id);


--
-- Name: axiologiseis fk_axiologisi; Type: FK CONSTRAINT; Schema: public; Owner:
postgres
--

ALTER TABLE ONLY public.axiologiseis
    ADD CONSTRAINT fk_axiologisi FOREIGN KEY (axiologisiid) REFERENCES
public.axiologisi(id);


--
-- Name: axiologiseis fk_axiologitisorg; Type: FK CONSTRAINT; Schema: public;
Owner: postgres
--

ALTER TABLE ONLY public.axiologiseis
    ADD CONSTRAINT fk_axiologitisorg FOREIGN KEY (axiologitisorgid) REFERENCES
public.organismos(id);


--
-- Name: epist_pedio_ergou fk_epist_pedio; Type: FK CONSTRAINT; Schema: public;
Owner: postgres
--

ALTER TABLE ONLY public.epist_pedio_ergou
    ADD CONSTRAINT fk_epist_pedio FOREIGN KEY (epist_pedio_onoma) REFERENCES
public.epist_pedio(onoma);


--
```

```
-- Name: ergazetai_se_ergo fk_erevnitis; Type: FK CONSTRAINT; Schema: public;
Owner: postgres
--

ALTER TABLE ONLY public.ergazetai_se_ergo
    ADD CONSTRAINT fk_erevnitis FOREIGN KEY (erevnitisid) REFERENCES
public.erevnitis(id);


--
-- Name: axiologiseis fk_erevnitis; Type: FK CONSTRAINT; Schema: public; Owner:
postgres
--

ALTER TABLE ONLY public.axiologiseis
    ADD CONSTRAINT fk_erevnitis FOREIGN KEY (erevnitisid) REFERENCES
public.erevnitis(id);


--
-- Name: ergazetai_se_ergo fk_ergo; Type: FK CONSTRAINT; Schema: public; Owner:
postgres
--

ALTER TABLE ONLY public.ergazetai_se_ergo
    ADD CONSTRAINT fk_ergo FOREIGN KEY (ergoid) REFERENCES public.ergo(id);


--
-- Name: epist_pedio_ergou fk_ergo; Type: FK CONSTRAINT; Schema: public; Owner:
postgres
--

ALTER TABLE ONLY public.epist_pedio_ergou
    ADD CONSTRAINT fk_ergo FOREIGN KEY (ergoid) REFERENCES public.ergo(id);


--
-- Name: paradoteo fk_ergo; Type: FK CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.paradoteo
    ADD CONSTRAINT fk_ergo FOREIGN KEY (ergoid) REFERENCES public.ergo(id);


--
-- Name: axiologiseis fk_ergo; Type: FK CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.axiologiseis
```

```
    ADD CONSTRAINT fk_ergo FOREIGN KEY (ergoid) REFERENCES public.ergo(id);


--
-- Name: axiologiseis fk_ergoorg; Type: FK CONSTRAINT; Schema: public; Owner:
postgres
--

ALTER TABLE ONLY public.axiologiseis
    ADD CONSTRAINT fk_ergoorg FOREIGN KEY (ergoorgid) REFERENCES
public.organismos(id);


--
-- Name: ergo fk_org; Type: FK CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.ergo
    ADD CONSTRAINT fk_org FOREIGN KEY (orgid) REFERENCES public.organismos(id);


--
-- Name: erevnitis fk_org; Type: FK CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.erevnitis
    ADD CONSTRAINT fk_org FOREIGN KEY (orgid) REFERENCES public.organismos(id);


--
-- Name: thlefwno fk_org; Type: FK CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.thlefwno
    ADD CONSTRAINT fk_org FOREIGN KEY (orgid) REFERENCES public.organismos(id);


--
-- Name: ergo fk_programmaonoma; Type: FK CONSTRAINT; Schema: public; Owner:
postgres
--

ALTER TABLE ONLY public.ergo
    ADD CONSTRAINT fk_programmaonoma FOREIGN KEY (programmaonoma) REFERENCES
public.programma(onoma);


--
-- Name: ergo fk_stelexosid; Type: FK CONSTRAINT; Schema: public; Owner: postgres
--
```

```
ALTER TABLE ONLY public.ergo
    ADD CONSTRAINT fk_stelexosid FOREIGN KEY (stelexosid) REFERENCES
public.stelexos(id);


--
-- PostgreSQL database dump complete
--
```

2.3 Οδηγίες Εγκατάστασης

Για την δημιουργία της εφαρμογής χρησιμοποιήθηκαν οι παρακάτω γλώσσες:

- HTML
- CSS
- JavaScript

Για την δημιουργία περιβάλλοντος χρήστη:

- Node.js (στον πάροχο gatsbyjs) για την δημιουργία του εξυπηρετητή ερωτημάτων SQL.
- PostgreSQL για την υλοποίηση της βάσης.
- Οι φόρμες εισαγωγής δεδομένων σχεδιάστηκαν και υλοποιήθηκαν μέσω της ιστοσελίδας cognitoforms.com.

Οι βιβλιοθήκες (για την ομιλία με την βάση δεδομένων και την εμφάνιση των φορμών) και τα configurations βρίσκονται στο GitLab repository με όνομα **gatsby-config.js**.

Οδηγίες για την εγκατάσταση της εφαρμογής:

Το σύστημα της βάσης και η εισαγωγή δεδομένων γίνονται μέσω τερματικού

1. Για τοπική χρήση χρειάζονται:
   α. Εξυπηρετητής Postgres
   β. Πρόγραμμα πελάτη

2. Χειρωνακτική εκκίνηση του εξυπηρετητή αν δεν έχει ξεκινήσει αυτόματα.

3. Εκτέλεση εντολής:
   sudo -u postgres psql postgres < create_tables.sql
   για την δημιουργία των πινάκων.

4. Εκτέλεση εντολής:
   sudo -u postgres psql postgres < insert_data.sql
   για την εισαγωγή των στοιχείων των πινάκων

5. Η διεύθυνση του εξυπηρετητή Postgres είναι:

postgres://oxcgoahw:L3qnvQ3bxno8auns5HN5w2nOgBjGXRW@tyke.db.elephantsql.com/oxcgoahw

η οποία δεν είναι αποθηκευμένη στο κώδικα git για λόγους ασφαλείας, αλλά είναι ρύθμιση που αποθηκεύεται στον πάροχο ιστοσελίδων.

Το πρόγραμμα node.js που την χρειάζεται την διαβάζει από μεταβλητή περιβάλλοντος (environment variable).

Υπόμνημα:
Οι δοκιμές-μικροαλλαγές της βάσης μπορούν να γίνουν μέσω φυλλομετρητή από την κονσόλα του ElephantSQL.

6. Για την είσοδο στην εφαρμογή μεταβαίνουμε στην ιστοσελίδα:

https://dblab2022.gtsb.io/menu.html