Laboratorio

Corso di Basi di Dati e Web

A.A. 2019-2020

Progetto “Catena di supermercati”

Gruppo composto da:

Margherita Pindaro margherita.pindaro@studenti.unimi.it

Data di consegna: *27/07/2020*

Studenti che intendono partecipare alla prova:

Margherita Pindaro

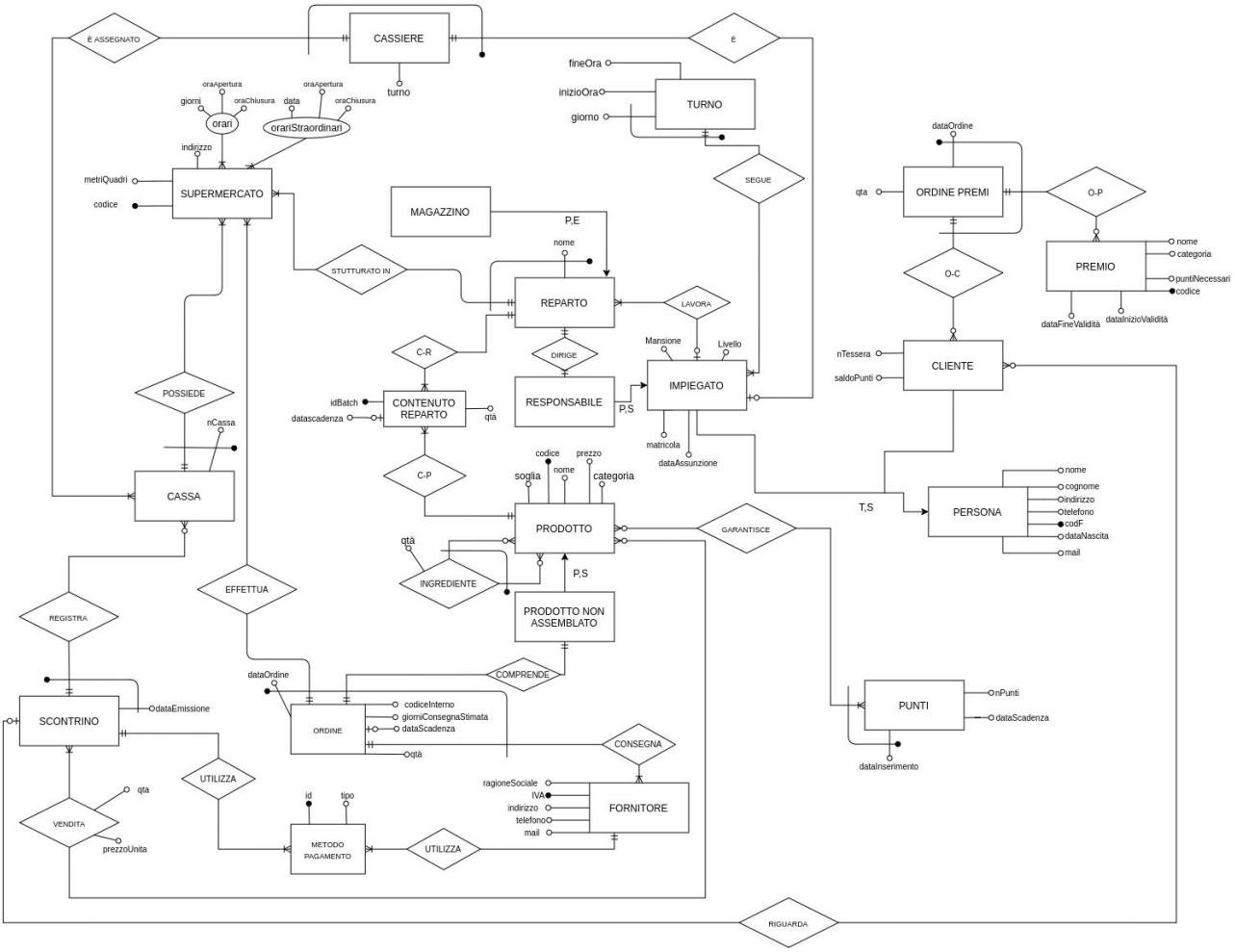
Il progetto è già stato consegnato in precedenza? No

1. Progettazione concettuale

1.1 Schema ER

Prima di realizzare lo schema ER ho formulato alcune ipotesi aggiuntive:

- I cassieri, in quanto impiegati, hanno una mansione (che corrisponderà a “Cassiere”) e un livello.
- Durante una vendita sono venduti i prodotti presenti nel reparto con data di scadenza più vicina.
- È sempre possibile capire in che reparto è stato comperato un prodotto perchè, per ogni reparto di ogni supermercato, deve esserci, per ogni prodotto, almeno un tupla dentro alla relazione *CONTENUTO REPARTO*, anche se la quantità è a zero. In tal caso la tupla rappresenterà lo scaffale vuoto. Per evitare la “duplicazione” di scaffali vuoti per lo stesso prodotto si prevede che quando, in seguito a un update, un batch avrà *qta* = 0, allora un trigger si occuperà di eliminare la tupla se ci sono altre tuple a rappresentare lo scaffale vuoto per quel prodotto.
- I magazzini dei supermercati, avendo comunque persone che vi lavorano al suo interno, è considerabile un reparto.



A questo link è possibile visualizzarlo meglio

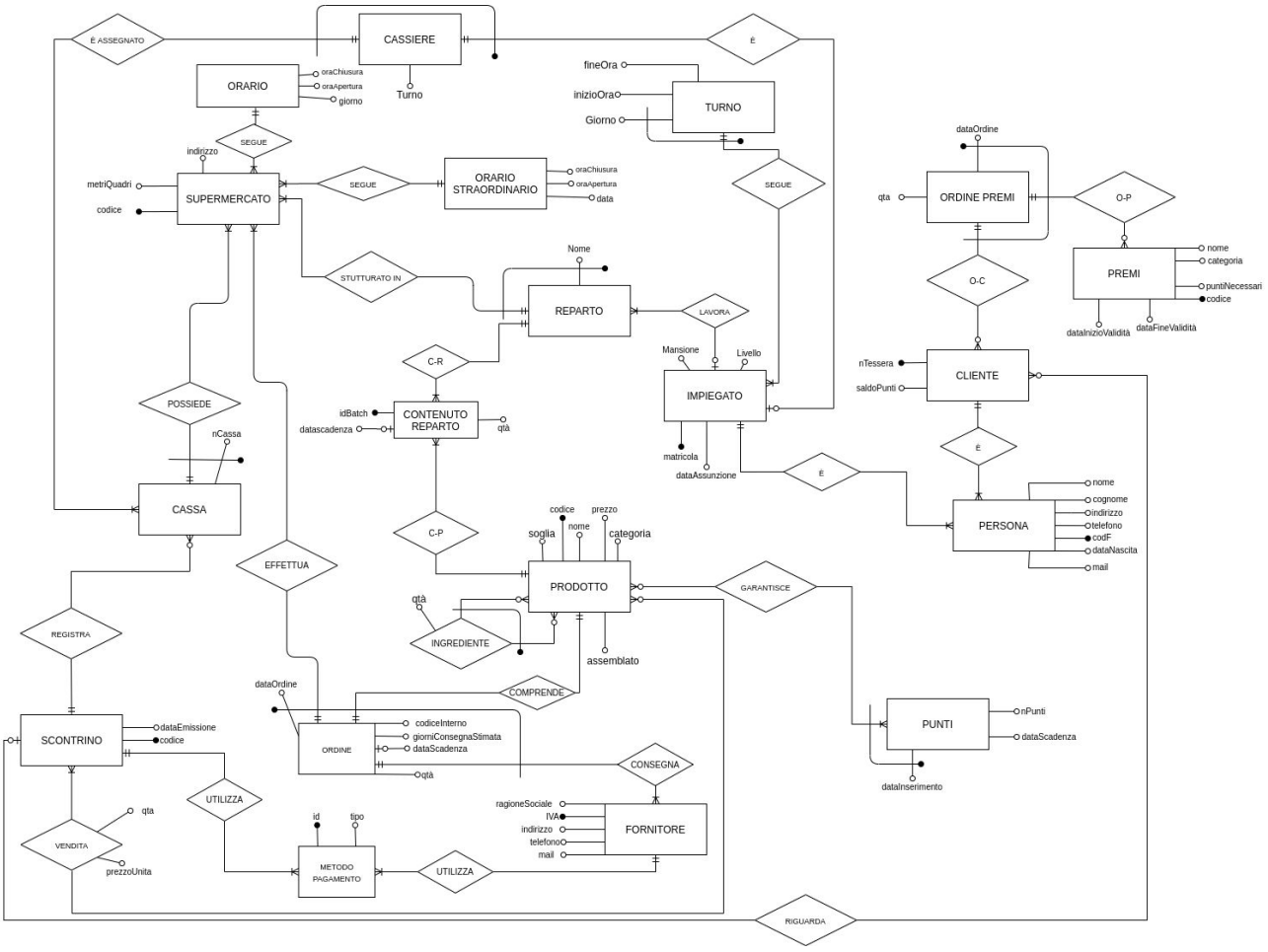
1.2 Vincoli di dominio

1. Ogni coppia *responsabile-reparto* deve partecipare alla relazione *LAVORA.*
2. *Mail* in *PERSONA* è unica e *dataNascita* non deve essere una data futura.

3. In *CONTENUTO REPARTO dataScadenza*, *reparto* e *prodotto* sono unici, mentre *qta* è maggiore o uguale a zero.
4. In *Cliente saldoPunti* deve essere positivo o uguale a zero.
5. In *PUNTI*, *punti* deve essere strettamente positivo e la *dataScadenza* deve essere maggiore di *dataInserimento*. Inoltre, quando viene inserito un nuovo record, se si riferisce a un prodotto che già è presente in *PUNTI*, allora *dataInserimento* nel nuovo record deve essere maggiore di *dataScadenza* del vecchio record.
6. In *ORDINEPREMI*, *qta* deve essere strettamente positivo e *dataOrdine* compresa tra *dataInizioValidità* e *dataFineValidità*.
7. In *IMPIEGATO*, *livello* deve essere compreso tra uno e sette e *dataAssunzione* non può essere una data futura.
8. In *VENDITE, qta* deve essere strettamente maggiore di zero e il *prezzoUnità* non negativo.
9. In *ORDINE*, *qta* deve essere strettamente maggiore di zero e, se è presente, *dataScadenza* deve essere una data futura. *giorniConsegnaStimata*, che rappresenta il tempo di spedizione espresso in giorni, deve essere maggiore o uguale a zero. Inoltre *dataOrdine* non deve essere una data futura.
10. Negli attributi composti *orari* e *orariStraordinari*, *oraChiusura* deve essere maggiore di *oraApertura*
11. In *TURNI*, *giorno* può solo assumere i seguenti valori: *Lunedì, Martedì, Mercoledì, Giovedì, Venerdì, Sabato, Domenica*.
12. In *INGREDIENTE qta* deve essere strettamente positivo.
13. In *PREMI*, *puntiNecessari* deve essere strettamente positivo *dataInizioValidità* minore di *dataFineValidità.*
14. *In PRODOTTO soglia* deve essere strettamente positivo.
15. In *SCONTRINI dataEmissione* deve essere una data non futura.

2. Progettazione logica

2.1 Schema ER ristrutturato



A questo link è possibile visualizzarlo meglio

2.2 Vincoli di dominio

1. Ogni coppia *responsabile-reparto* deve partecipare alla relazione *LAVORA.*
2. *Mail* in *PERSONA* è unica e *dataNascita* non deve essere una data futura.
3. In *CONTENUTO REPARTO dataScadenza*, *reparto* e *prodotto* sono unici, mentre *qta* è maggiore o uguale a zero.
4. In *Cliente saldoPunti* deve essere positivo o uguale a zero.
5. In *PUNTI*, *punti* deve essere strettamente positivo e la *dataScadenza* deve essere maggiore di *dataInserimento*. Inoltre, quando viene inserito un nuovo record, se si riferisce a un prodotto che già è presente in *PUNTI*, allora *dataInserimento* nel nuovo record deve essere maggiore di *dataScadenza* del vecchio record.
6. In *ORDINEPREMI*, *qta* deve essere strettamente positivo e *dataOrdine* compresa tra *dataInizioValidità* e *dataFineValidità*.
7. In *IMPIEGATO*, *livello* deve essere compreso tra uno e sette e *dataAssunzione* non può essere una data futura.
8. In *VENDITE, qta* deve essere strettamente maggiore di zero e il *prezzoUnità* non negativo.
9. In *ORDINE*, *qta* deve essere strettamente maggiore di zero e, se è presente, *dataScadenza* deve essere una data futura. *giorniConsegnaStimata*, che

rappresenta il tempo di spedizione espresso in giorni, deve essere maggiore o uguale a zero. Inoltre *dataOrdine* non deve essere una data futura.

10. Negli attributi composti *orari* e *orariStraordinari*, *oraChiusura* deve essere maggiore di *oraApertura*
11. In *TURNI*, *giorno* può solo assumere i seguenti valori: *Lunedì, Martedì, Mercoledì, Giovedì, Venerdì, Sabato, Domenica*.
12. In *INGREDIENTE qta* deve essere strettamente positivo.
13. In *PREMI*, *puntiNecessari* deve essere strettamente positivo *dataInizioValidità* minore di *dataFineValidità.*
14. In *PRODOTTO soglia* deve essere strettamente positivo.
15. In *SCONTRINI dataEmissione* deve essere una data non futura.

Quelli aggiunti in seguito alla ristrutturazione sono:

16. I prodotti che partecipano a *CONTENUTO REPARTO* devono avere attributo *assemblato* false, mentre invece i prodotti che partecipano a *INGREDIENTE* nei termini come prodotto che ha determinati ingredienti, allora *assemblato* deve essere true.
17. Per ogni supermercato deve esserci uno ed un solo reparto denominato “Magazzino”
18. Per ogni reparto di ogni supermercato in *IMPIEGATO* deve esserci una e una sola tupla con *Mansione* corrispondente a “Responsabile”.

2.3 Modello relazionale

- Casse(*supermercato*, nCassa, *addetto*)
- Cassieri(*impiegato, supermercatoCassa, nCassa,* turno)
- Clienti(nTessera, saldoPunti, *persona*)
- ContenutoReparto(idBatch, *supermercatoReparto*, *nomeReparto*, prodotto, dataScadenza, qta)
- Fornitori(iva, ragioneSociale, indirizzo, telefono, mail, *pagamento*)
- Impiegati(matricola, dataAssunzione, mansione, livello, *supermercato, nomeReparto, persona*)
- Ingredienti(*ingrediente, prodottoFinale*, qta)
- MetodoPagamento(id, tipo)
- Orari(*supermercato*, giorni, oraApertura, oraChiusura)
- OrariStraordinari(*supermercato*, data, oraApertura, oraChiusura)
- Ordini(codiceInterno, dataOrdine, qta, *fornitore*, *prodotto, supermercato*, dataScadenza, giorniConsegnaStimata)
- OrdiniPremi(*premio*, *cliente,* dataOrdine, qta)
- Persone(codF, nome, cognome, indirizzo, telefono, dataNascita, mail)
- Premi(codice, nome, categoria, puntiNecessari, dataInizioValidita, dataFineValidita)
- Prodotti(id, nome, prezzo, categoria, soglia, assemblato)
- Punti(*prodotto*, dataInserimento, dataScadenza, nPunti)
- Reparti(*supermercato*, nome)
- Scontrini(id, dataEmissione, *pagamento*, *nCassa, supermercatoCassa, cliente*)
- Supermercati(codice, metriquadri, indirizzo)
- Turni(*impiegato, giorno*, inizioOra, fineOra)
- Vendite(*scontrino, prodotto,* qta, prezzoUnita)

2.4 Codice SQL

Riportare il codice SQL di creazione e alterazione delle tabelle per attuare i vincoli individuati (in particolare vincoli di primary e foreign key, not null, unique e check).

```
create table if not exists public.premi
(
  codice bigint not null
     constraint premi_pk
        primary key,
  nome varchar not null,
  categoria varchar not null,
  "puntiNecessari" integer not null
     constraint puntinecessariperunpremiomaggioridi0
        check ("puntiNecessari" > 0),
  "dataInizioVadilita" date not null,
  "dataFineValidita" date not null,
  constraint validitàpremioinunintervalloesistente
     check ("dataInizioVadilita" < "dataFineValidita")
);

create table if not exists public.persone
(
  "codF" varchar(16) not null
     constraint persone_pk
        primary key,
  nome varchar(20) not null,
  cognome varchar(20) not null,
  indirizzo varchar not null,
  telefono varchar(12) not null,
  "dataNascita" date not null,
  mail varchar not null
);

create unique index if not exists persone_mail_uindex
  on public.persone (mail);

create table if not exists public.supermercati
(
  codice varchar not null
     constraint supermercato_pk
        primary key,
  metriquadri double precision not null,
  indirizzo varchar
);

create table if not exists public.prodotti
(
  id bigint not null
     constraint prodotti_pk
        primary key,
  nome varchar not null,
  prezzo double precision not null,
  categoria varchar not null,
  assemblato boolean default false not null,
  soglia integer not null
     constraint soglianonnegativa
        check (soglia >= 1)
);

create table if not exists public."metodoPagamento"
```

```
(
  id serial not null
     constraint metodopagamento_pk
        primary key,
  tipo varchar not null
);

create table if not exists public.fornitori
(
  iva varchar(11) not null
     constraint fornitore_pk
        primary key,
  "ragioneSociale" varchar not null,
  indirizzo varchar not null,
  telefono varchar(12) not null,
  mail varchar not null,
  pagamento integer not null
     constraint "metodoPagamentoFornitore"
        references public."metodoPagamento"
           on update cascade on delete cascade
);

create table if not exists public.clienti
(
  "nTessera" varchar not null
     constraint clienti_pk
        primary key,
  "saldoPunti" integer not null
     constraint puntipositivi
        check ("saldoPunti" >= 0),
  persona varchar(16) not null
     constraint clienti_persone_codf_fk
        references public.persone
           on update cascade on delete cascade
);

create unique index if not exists clienti_persona_uindex
  on public.clienti (persona);

create table if not exists public.orari
(
  giorni varchar not null,
  supermercato varchar not null
     constraint orari_supermercato_indirizzo_fk
        references public.supermercati
           on update cascade on delete cascade,
  "oraApertura" time not null,
  "oraChiusura" time not null,
  constraint orari_pk
     primary key (giorni, supermercato),
  constraint primaaprepoichiude
     check ("oraChiusura" > "oraApertura")
);

create table if not exists public."orariStraordinari"
(
  supermercato varchar not null
     constraint oraristraordinari_supermercato_indirizzo_fk
        references public.supermercati
           on update cascade on delete cascade,
  data varchar not null,
  "oraApertura" time not null,
  "oraChiusura" time not null,
```

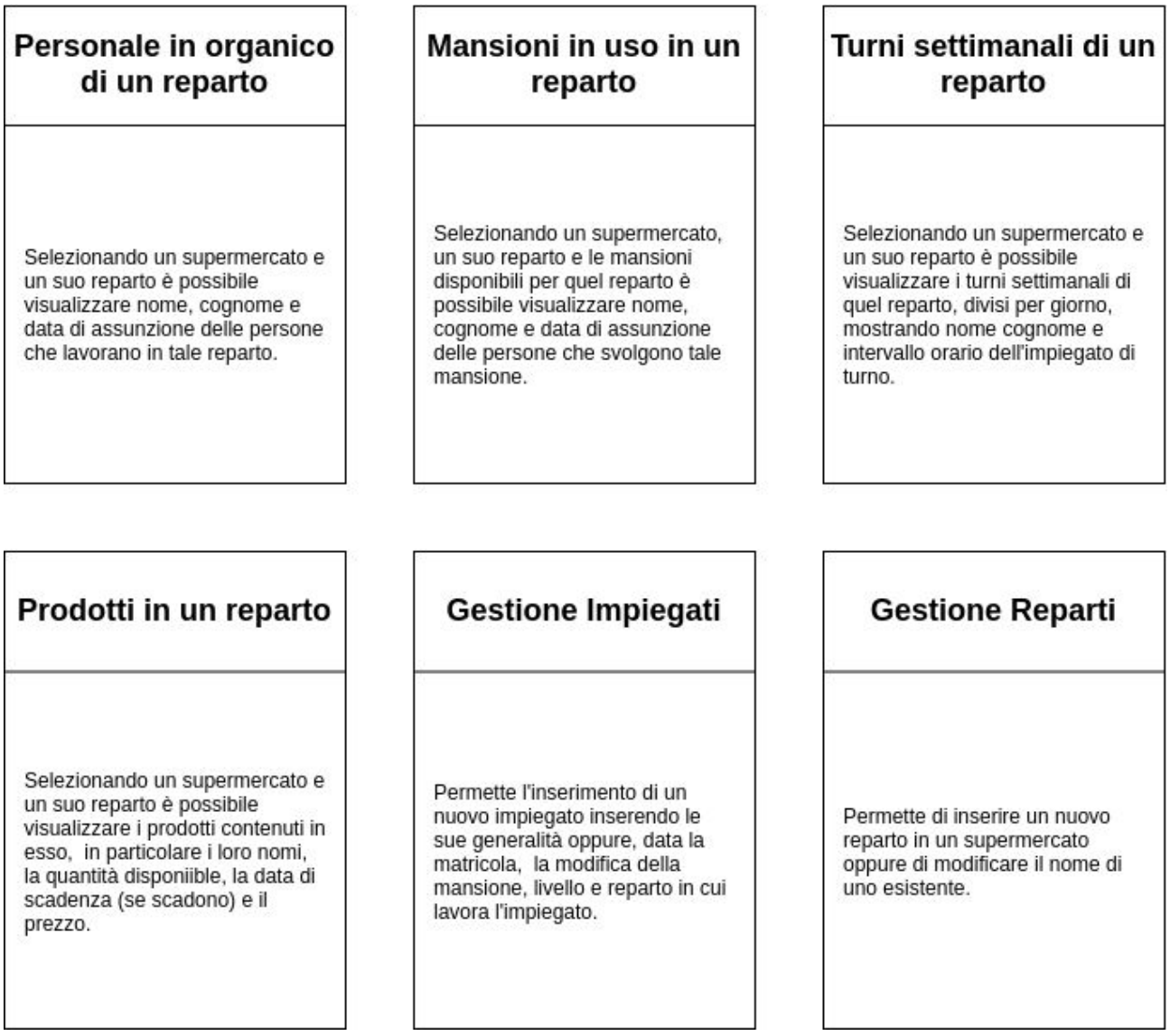
```
  constraint oraristraordinari_pk
     primary key (supermercato, data),
  constraint primaaprepoichiudestraordinario
     check ("oraChiusura" > "oraApertura")
);

create table if not exists public.ingredienti
(
  ingrediente integer not null
     constraint ingredienti_prodotti_id_fk
        references public.prodotti
           on update cascade on delete cascade,
  "prodottoFinale" integer not null
     constraint ingredienti_prodotti_id_fk_2
        references public.prodotti
           on update cascade on delete cascade,
  qta integer not null
     constraint stimatempogiornipositiva
        check (qta > 0),
  constraint table_name_pk
     primary key (ingrediente, "prodottoFinale")
);

create table if not exists public.punti
(
  prodotto integer not null
     constraint punti_prodotti_id_fk
        references public.prodotti
           on update cascade on delete cascade,
  "dataInserimento" date not null,
  "dataScadenza" date not null,
  "nPunti" integer not null
     constraint npuntichedaunprodottopositivi
        check ("nPunti" >= 1),
  constraint punti_pk
     primary key (prodotto, "dataInserimento"),
  constraint datainserimentoprodottopuntiprimadatascadenzapunti
     check ("dataInserimento" <= "dataScadenza")
);

create table if not exists public."ordiniPremi"
(
  premio integer not null
     constraint ordinepremi_premi_codice_fk
        references public.premi
           on update cascade on delete cascade,
  cliente varchar not null
     constraint ordinepremi_clienti_ntessera_fk
        references public.clienti
           on update cascade on delete cascade,
  "dataOrdine" date not null,
  qta integer not null
     constraint qtaordinepremiopositiva
        check (qta >= 1),
  constraint ordinepremi_pk
     primary key (premio, cliente, "dataOrdine")
);

create table if not exists public.reparti
(
  supermercato varchar not null
     constraint reparti_supermercato_indirizzo_fk
        references public.supermercati
```

```
            on update cascade on delete cascade,
  nome varchar not null,
  constraint reparti_pk
     primary key (supermercato, nome)
);

create table if not exists public.impiegati
(
  matricola serial not null
     constraint impiegati_pk
        primary key,
  "dataAssunzione" date not null
     constraint dataassunzionenonfutura
        check ("dataAssunzione" <= CURRENT_DATE),
  mansione varchar not null,
  livello integer not null
     constraint livelloimpiegato
        check ((livello >= 1) AND (livello <= 7)),
  supermercato varchar,
  "nomeReparto" varchar,
  persona varchar(16) not null
     constraint impiegati_persone_codf_fk
        references public.persone
           on update cascade on delete cascade,
  constraint impiegati_reparti_supermercato_nome_fk
     foreign key (supermercato, "nomeReparto") references public.reparti
        on update cascade on delete cascade
);

create unique index if not exists impiegati_persona_uindex
  on public.impiegati (persona);

create table if not exists public.casse
(
  supermercato varchar not null
     constraint casse_supermercato_indirizzo_fk
        references public.supermercati
           on update cascade on delete cascade,
  "nCassa" integer not null,
  constraint casse_pk
     primary key (supermercato, "nCassa")
);

create table if not exists public.scontrini
(
  id integer not null
     constraint scontrini_pk
        primary key,
  "dataEmissione" date not null
     constraint scontrininonnelfuturo
        check ("dataEmissione" <= CURRENT_DATE),
  pagamento integer not null
     constraint scontrini_metodopagamento_id_fk
        references public."metodoPagamento"
           on update cascade on delete cascade,
  "nCassa" integer not null,
  "supermercatoCassa" varchar not null,
  cliente varchar
     constraint scontrini_clienti_ntessera_fk
        references public.clienti
           on update cascade on delete cascade,
  constraint scontrini_casse_supermercato_ncassa_fk
     foreign key ("supermercatoCassa", "nCassa") references public.casse
```

```
        on update cascade on delete cascade
);

create table if not exists public.vendite
(
  scontrino integer not null
     constraint vendite_scontrini_id_fk
        references public.scontrini
           on update cascade on delete cascade,
  prodotto integer not null
     constraint vendite_prodotti_id_fk
        references public.prodotti
           on update cascade on delete cascade,
  qta integer not null
     constraint vendutoalmenounprodtto
        check (qta > 0),
  "prezzoUnità" double precision not null
     constraint prezzovenditaminimo0
        check ("prezzoUnità" >= (0)::double precision),
  constraint vendite_pk
     primary key (scontrino, prodotto)
);

create table if not exists public."contenutoReparto"
(
  "idBatch" integer not null
     constraint contenutoreparto_pk
        primary key,
  "supermercatoReparto" varchar not null,
  "nomeReparto" varchar not null,
  prodotto integer not null
     constraint contenutoreparto_prodotti_id_fk
        references public.prodotti
           on update cascade on delete cascade,
  "dataScadenza" date,
  qta integer not null
     constraint qtacontenutoinunreparto
        check (qta >= 0),
  constraint contenutoreparto_reparti_supermercato_nome_fk
     foreign key ("supermercatoReparto", "nomeReparto") references public.reparti
        on update cascade on delete cascade
);

create unique index if not exists
contenutoreparto_prodotto_datascadenza_nomereparto_supermercato
     on public."contenutoReparto" (prodotto, "dataScadenza", "nomeReparto",
"supermercatoReparto");

create table if not exists public.ordini
(
  "codiceInterno" integer not null,
  "dataOrdine" date not null
     constraint dataordinenonfutura
        check ("dataOrdine" <= CURRENT_DATE),
  qta integer not null
     constraint qtaordinestrettamentepositiva
        check (qta > 0),
  fornitore varchar(16) not null
     constraint ordini_fornitore_iva_fk
        references public.fornitori
           on update cascade on delete cascade,
  prodotto integer not null
     constraint ordini_prodotti_id_fk
```

```
        references public.prodotti
           on update cascade on delete cascade,
  supermercato varchar not null
     constraint ordini_supermercato_indirizzo_fk
        references public.supermercati
           on update cascade on delete cascade,
  "dataScadenza" date
     constraint datascadenzafutura
        check ("dataScadenza" >= CURRENT_DATE),
  "giorniConsegnaStimata" integer not null
     constraint stimatempogiornipositiva
        check ("giorniConsegnaStimata" >= 0),
  constraint ordini_pk
     primary key ("dataOrdine", fornitore, prodotto, supermercato)
);

create table if not exists public.turni
(
  impiegato integer not null
     constraint turni_impiegati_matricola_fk
        references public.impiegati
           on update cascade on delete cascade,
  giorno varchar not null
     constraint eungiornodellasettimana
              check  (((giorno)::text  =  'Lunedi'::text)  OR  ((giorno)::text  =
'Martedi'::text)  OR  ((giorno)::text  =  'Mercoledi'::text)  OR  ((giorno)::text  =
'Giovedi'::text)  OR  ((giorno)::text  =  'Venerdi'::text)  OR  ((giorno)::text  =
'Sabato'::text) OR ((giorno)::text = 'Domenica'::text)),
  "inizioOra" time not null,
  "fineOra" time not null,
  constraint turni_pk
     primary key (impiegato, giorno)
);

create table if not exists public.cassieri
(
  impiegato integer not null
     constraint cassieri_impiegati_matricola_fk
        references public.impiegati
           on update cascade on delete cascade,
  "supermercatoCassa" varchar not null,
  "nCassa" integer not null,
  turno varchar not null,
  constraint cassieri_pk
     primary key (impiegato, "supermercatoCassa", "nCassa"),
  constraint cassieri_casse_supermercato_ncassa_fk
     foreign key ("supermercatoCassa", "nCassa") references public.casse
        on update cascade on delete cascade
);
```

3. Progettazione del sito



Nome del database PostgreSQL: CatenaDiSupermercati