

Esame del 12/01/2026 – Turno Unico

Si consideri il database “formula1”, contenente informazioni su tutte le gare, i costruttori, i piloti ed i circuiti di Formula 1, estratto dai dati pubblicati sul sito <http://ergast.com/mrd/>. Il database è strutturato secondo il diagramma ER della pagina seguente.

Si intende costruire un’applicazione che permetta di interrogare tale base dati, e calcolare informazioni a proposito delle gare disputate.

L’applicazione dovrà svolgere le seguenti funzioni:

PUNTO 1

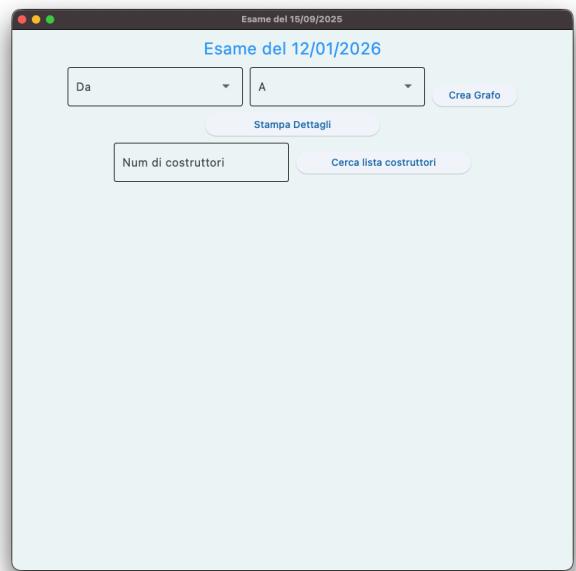
- 1) L’utente seleziona da due menù a tendina due anni che definiscono un range di interesse. I menù dovranno essere riempiti interrogando il database per ottenere gli anni in cui è stato disputato il campionato.
- 2) Premendo sul tasto *Crea grafo*, l’applicazione dovrà costruire un *grafo semplice e pesato*, così costituito:
 - a) I nodi sono costituiti da tutti i costruttori (*constructors*) che hanno partecipato ad almeno una gara nel corso del periodo selezionato nel punto 1), estremi inclusi. Si considerino solo le partecipazioni valide, ovvero dove almeno un pilota del costruttore ha correttamente tagliato il traguardo (campo *position* della tabella *results* NOT NULL).
 - b) Due nodi sono connessi da un arco se e solo se i due costruttori hanno condiviso almeno un pilota durante il periodo selezionato (campo *driverId* della tabella *results*). Un pilota si considera "condiviso" se ha corso per entrambi i costruttori in gare diverse all’interno del range di anni selezionato. Il peso dell’arco è pari al numero di piloti distinti che hanno corso per entrambi i costruttori nel periodo considerato. Si considerino solo le gare in cui il pilota ha correttamente tagliato il traguardo.
- 3) Costruito il grafo, l’applicazione visualizza il numero di nodi e di archi presenti nel grafo. Alla pressione del tasto "Stampa dettagli", il programma dovrà:
 - a) Stampare i tre archi di peso maggiore (le coppie di costruttori che hanno condiviso più piloti);
 - b) Stampare il numero di componenti connesse;
 - c) Identificare la componente connessa di dimensione maggiore, e stamparne tutti i nodi, ordinati in senso decrescente secondo il grado dei nodi.

PUNTO 2

Si vuole valutare la diversità anagrafica dei "veterani" (piloti più anziani) nelle diverse scuderie. Dato un valore K intero fornito dall’utente, l’obiettivo del programma è di identificare un set di K costruttori distinti tale per cui:

- a) Ciascun costruttore faccia parte di una componente connessa distinta, ovvero i costruttori selezionati non devono aver mai condiviso piloti tra loro;
- b) La differenza fra la data di nascita del pilota più anziano che ha corso per ciascun costruttore sia minima. In altre parole, per ogni costruttore si identifichi il "veterano" (il pilota con data di nascita più vecchia che ha corso per quel team nel periodo selezionato), e si minimizzi il range di età tra questi veterani.

SUGGERIMENTO: Per risolvere questo punto, una strategia è quella di scrivere un metodo del DAO che interroghi il db per ottenere la data di nascita (campo *dob* della tabella *drivers*) del pilota più anziano che ha corso per una certa

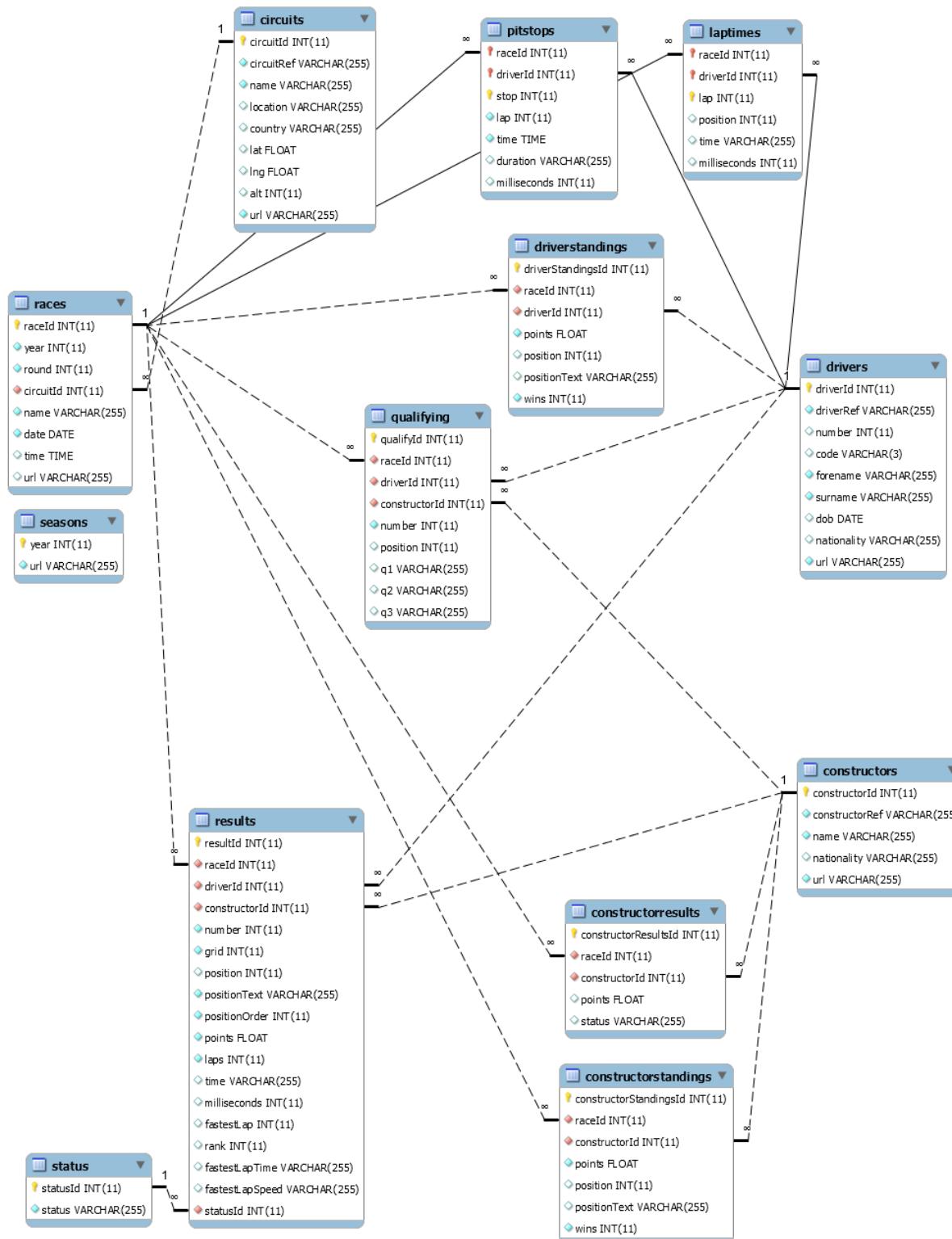


scuderia, e di utilizzare tale metodo per riempire il campo “oldest_driver_dob” della classe `Constructor` proposta nel codice fornito.

Si stampino la lista dei K costruttori selezionati, lo scarto di età in giorni, il costruttore con il veterano più giovane e quello con il veterano più anziano.

Nella realizzazione del codice, si lavori a partire dalle classi e dal database contenuti nel progetto di base. È ovviamente permesso aggiungere o modificare classi e metodi.

Tutti i possibili errori di immissione, validazione dati, accesso al database, ed algoritmici devono essere gestiti, non sono ammesse eccezioni generate dal programma.



ESEMPI

