TEAM 1 – Documentazione in modalità seconda prova

Alessio Trentin – Ion Cristian Donisan – Abreham Bologna

Progetto *Car Pooling*

DOMINIO

azienda, piattaforma, car pooling, viaggiatori, territorio, utenti, autisti, passeggeri, passaggio, macchina, generalità, numero patente di guida, scadenza, telefono, email, fotografia, viaggio, città partenza, città destinazione, tempo di percorrenza, prenotazioni, documento di identità, voto medio, giudizi, accettazione, rifiuto, promemoria.

DIAGRAMMA ORM

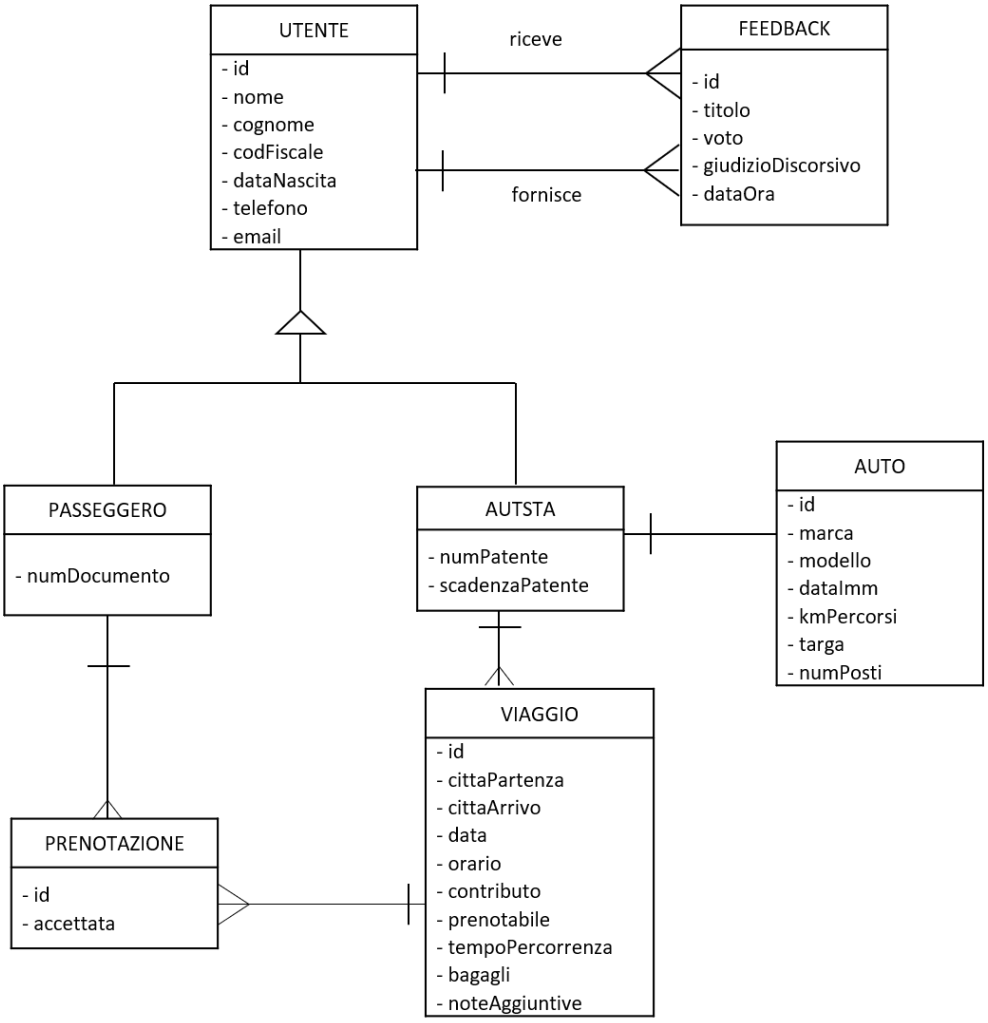
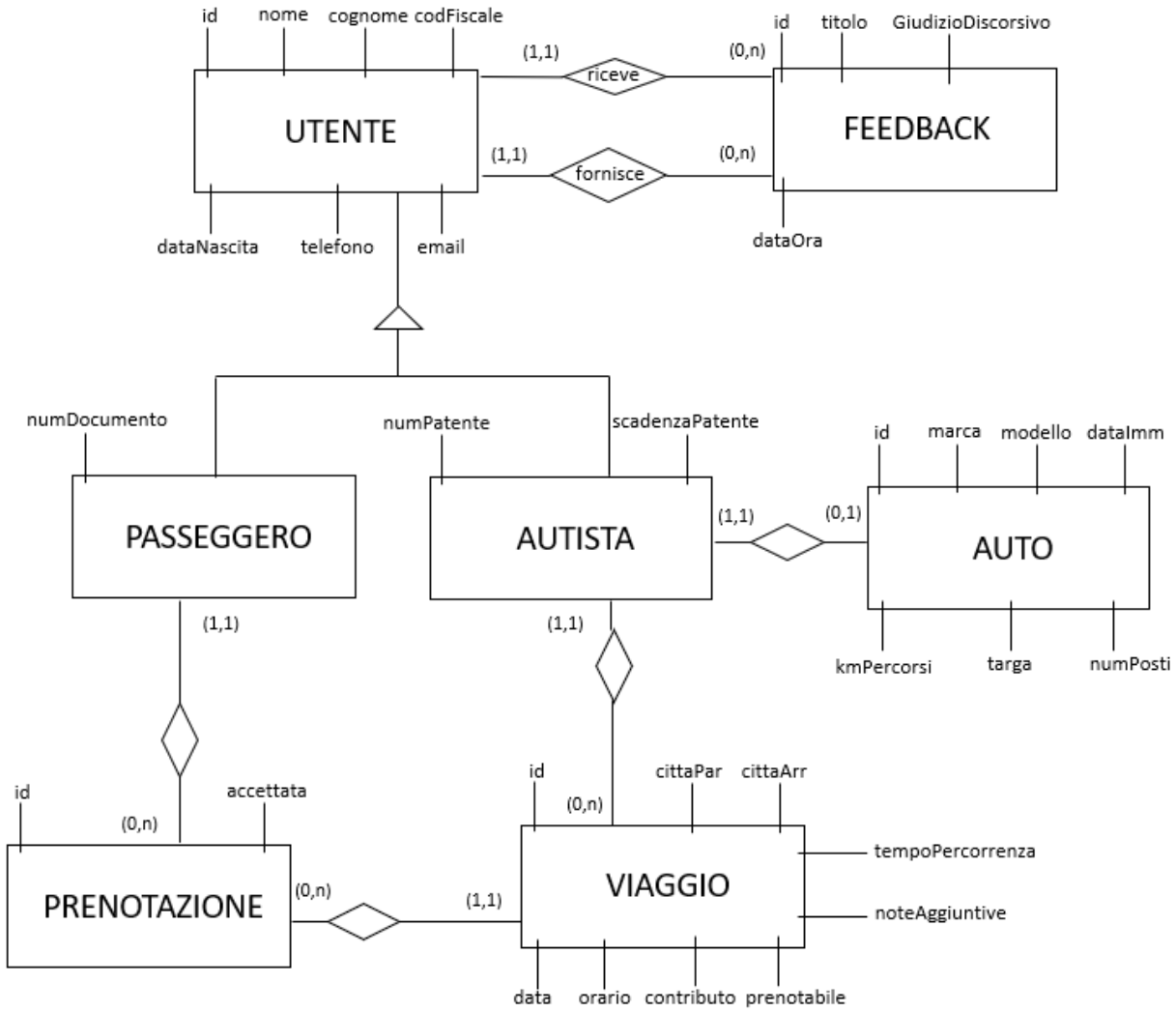


DIAGRAMMA ER



OSSERVAZIONI

#1 Viaggiatore è alias di Utente.

#2 per generalità di Utente si intende nome, cognome, codFiscale e dataNascita.

#3 telefono e email sono proprietà di Utente.

#4 numPatente e scadenzaPatente sono proprietà di Autista.

#5 Passaggio è alias di Viaggio.

#6 Macchina è alias di Auto.

#7 la fotografia dell'utente non è richiesta per semplicità di implementazione.

#8 città di destinazione è alias di cittaArrivo.

#9 cittaPartenza, cittaArrivo e tempo di percorrenza sono proprietà di Viaggio.

#10 prenotabile in Viaggio è un boolean che indica se il viaggio è ancora prenotabile.

#11 documento di identità è numDocumento in Passeggero.

#12 voto medio non è presente in Utente perché calcolabile.

#13 voto e giudizioDiscorsivo sono proprietà di Feedback.

#14 accettata in Prenotazione è un boolean che indica se la prenotazione è stata accettata.

#15 azienda, piattaforma, car pooling, territorio e promemoria non contengono informazioni.

#16 un'Auto può essere di un solo Autista.

#17 un'Autista può registrare solo un'Auto.

ASSOCIAZIONI

Associazione *Utente – Feedback riceve*

Un Utente può ricevere zero o più Feedback, un Feedback deve essere destinato ad un solo Utente.

Associazione *Utente – Feedback fornisce*

Un Utente può fornire zero o più Feedback, un Feedback deve essere fornito da un solo Utente.

Associazione *Autista – Auto*

Un Autista può avere al massimo un'Auto, un'Auto deve appartenere ad un solo Autista.

Associazione *Autista – Viaggio*

Un Autista può offrire zero o più Viaggi, un Viaggio è offerto da un solo Autista.

Associazione *Passeggero – Prenotazione*

Un Passeggero può effettuare diverse Prenotazioni, una Prenotazione può essere effettuata da un solo Passeggero.

Associazione *Prenotazione – Viaggio*

Un Viaggio può ricevere zero o più Prenotazioni, una Prenotazione è rivolta ad un solo viaggio.

SCHEMA LOGICO RELAZIONALE

UTENTE (id, nome, cognome, dataNascita, codFiscale, telefono, email, numDocumento, numPatente, scadenzaPatente, discriminatore)

Chiave candidata: codFiscale.

FEEDBACK (id, titolo, voto, giudizioDiscorsivo, dataOra, id_utente_destinatario, id_utente_autore)

Chiave candidata: (dataOra, id_utente_destinatario, id_utente_autore).

id_utente_destinatario: FK Utente (associazione riceve).

Id_utente_autore: FK Utente (associazione fornisce).

AUTO(id, marca, modello, datalmm, kmPercorsi, targa, numPosti, autista_id)

Chiave candidata: targa.

autista_id: FK Utente.

VIAGGIO (id, cittaPartenza, cittaArrivo, data, orario, contributo, prenotabile, tempoPercorrenza, bagagli, noteAggiuntive, autista_id)

Chiave candidtata: (data, orario, autista_id).

autista_id: FK Utente.

PRENOTAZIONE (id, accettata, passeggero_id, viaggio_id)

Chiave candidata: (passeggero_id, viaggio_id).

passeggero_id: FK Utente.

viaggio_id: FK Viaggio.

QUERY SQL

a) CREATE VIEW Autista (id) IN (
SELECT id
FROM Utente
WHERE numPatente != NULL)

SELECT a.*, m.*

FROM Viaggio v INNER JOIN Autista a ON v.autista_id = a.id INNER

JOIN Auto m ON m.autista_id = a.id

WHERE v.prenotabile = TRUE AND v.cittaPartenza = cittaPartenza AND v.cittaArrivo = cittaArrivo AND v.data = data

ORDER BY v.data

b) SELECT v.cittaPartenza, v.cittaArrivo, v.data, a.*, m.*
FROM Prenotazione p INNER JOIN Viaggio v ON p.viaggio_id = v.id
INNER JOIN Autista a ON v.autista_id = a.id INNERJOIN Auto m ON a.id = m.autista_id
WHERE p.id = id

c) CREATE VIEW Passeggero (id) IN (
SELECT id
FROM Utente
WHERE numPatente = NULL)
SELECT f.id_utente_destinatario, f.AVG(voto)
FROM Viaggio v INNER JOIN Prenotazione p ON v.id = p.viaggio_id
INNER JOIN Passeggero pass ON p.passeggero_id = pass.id INNER JOIN Feedback f ON f.id_utente_destinatario = pass.id
WHERE f.AVG(voto) > votoMedio
GROUP BY f.id_utente_destinatario