15420765101 11454232046 27401706422 75721021601 14307131652 43135317123 234115447451

COGNOME	NOME	MATRICOLA

Basi di Dati 2022/23 – 09 gennaio 2024

Closed book (non è possibile consultare materiale)

Tempo a disposizione: 1h 45' parte I e II [1h 20' se senza esercizio I.A] + 45' parte III

Esercizio I.A REVERSE ENGINEERING * gli studenti che hanno aderito a opzione 2 sono esonerati

Si consideri il seguente schema relazionale, relativo a noleggi di autoveicoli

AGENZIA (NomeAg, Indirizzo, Tel, Città, Stato)
VEICOLO (Targa, CategoriaV, Modello, Marca, Alimentazione, DataImm)
CLIENTE (IdCli, Nome, Indirizzo, Tel, *Email*, DataN, *NumeroPat*, ScadenzaPat)
TARIFFA (IdT, CategoriaV, TipoTar, ImportoPeriodo, ImportoKm)
ASSIC (IdCop, TipoCopertura, Descrizione, ImportoG, Massimale, Franchigia)
PRENOTAZIONE (IdP, *Targa*$^{\text{VEICOLO}}$, *DataRitiro*, DataConsegna, AgRitiro$^{\text{AGENZIA}}$, AgConsegna$^{\text{AGENZIA}}$, IdCli$^{\text{LIENTE}}$, IdT$^{\text{TARIFFA}}$, IDCop$^{\text{ASSIC}}$, Stato, Importo)
NOLEGGIO (IdP$^{\text{PRENOTAZIONE}}$, KmIniziali, KmFinali, Carburante, Costo)
GUIDATORE_AGG (IdP$^{\text{PRENOTAZIONE}}$, IdCli$^{\text{CLIENTE}}$)

1. si proponga uno schema concettuale Entity Relationship la cui traduzione dia luogo a tale schema logico

2. si modifichi lo schema in 1. per gestire gli eventuali incidenti avvenuti durante un noleggio, con informazioni legate ai danni subiti/causati e responsabilità.

COGNOME	NOME	MATRICOLA

Esercizio I.B NORMALIZZAZIONE

1. Si consideri la seguente relazione contenente informazioni su esami medici.

ESAME (CodiceFiscalePaziente, NomePaziente, CognomePaziente, DataNascitaPaziente, DataEsame, TipoEsame, Patologia, Apparecchio, TipoApparecchio, Dottore, Infermiere, Laboratorio, ASL)

Determinare, per ciascuna delle seguenti affermazioni, se rappresentano dipendenze funzionali per la relazione ESAME e, in caso affermativo, presentare la dipendenza:

a) Ogni apparecchio ha un unico tipo e si trova in un certo laboratorio.

APPARECCHIO ⟶ TIPOAPPARECCHIO, LABORATORIO

b) In ogni laboratorio possono lavorare diversi dottori e diversi infermieri.

NON È UNA DIPENDENZA FUNZIONALE IN QUANTO SAREBBE RELAZIONE UNO A MOLTI. OGNI LABORATORIO DEVE AVERE UN CERTO DOTTORE E UN CERTO INFERMIERE

c) Ogni dottore e ogni infermiere in ogni data lavorano in un certo laboratorio.

DOTTORE, INFERMIERE, DATAESAME ⟶ LABORATORIO

2. Data la relazione R(A,B,C,D,E) e le dipendenze funzionali E → AB, BC → D e A → C, determinare le chiavi di R e specificare se R è in 3NF o in BCNF, motivando le risposte.

CHIAVE CANDIDATA: E IN QUANTO NON APPARE A DX DELLA FRECCIA IN OGNI DIPENDENZA FUNZIONALE. CALCOLIAMO LA SUA CHIUSURA: $\{E\}^+ = \{E, A, B, C, D\}$. SICCOME ABBIAMO RICAVATO LA RELAZIONE R ALLORA E È UNICA CHIAVE DI R.

È IN 3NF? NO, NON ABBIAMO SUPERCHIAVI A SX DELLA FRECCIA E NON ABBIAMO ATTRIBUTI PRIMI A DX.

È IN BCNF? NO, NON ABBIAMO SUPERCHIAVI A SX DELLE FRECCE

COGNOME	NOME	MATRICOLA

Esercizio II.A – ALGEBRA RELAZIONALE

In riferimento al seguente schema relazionale (semplificazione dello schema dell'esercizio I)

AGENZIA (NomeAg, Indirizzo, Tel, Città, Stato)
VEICOLO (Targa, CategoriaV, Modello, Marca, Alimentazione, DataImm)
CLIENTE (IdCli, Nome, Indirizzo, Tel, *Email*, DataN, *NumeroPat*, ScadenzaPat)
PRENOTAZIONE (IdP, *Targa*[VEICOLO], *DataRitiro*, DataConsegna, AgRitiro[AGENZIA], AgConsegna[AGENZIA], IdCli[LIENTE], Stato, Importo)
NOLEGGIO (IdP[PRENOTAZIONE], KmIniziali, KmFinali, Carburante, Costo)
GUIDATORE_AGG (IdP[PRENOTAZIONE], IdCli[CLIENTE])

Formulare le seguenti interrogazioni in **algebra relazionale**.

1. Determinare categoria, marca e modello di veicoli prenotati con agenzia di ritiro e consegna diverse da clienti con meno di 21 anni.

$M_{CATEGORIAV,\ MODELLO,\ MARCA}\left(VEICOLO \bowtie \delta_{AGRITIRO \neq AGCONSEGNA}\left(PRENOTAZIONE\right) \bowtie \delta_{CURRENT\_DATE - DATAN < 21}\left(CLIENTE\right)\right)$

3. ~~Determinare l'email dei clienti~~ che hanno effettuato due prenotazioni diverse con la stessa data ritiro e agenzie di ritiro in città diverse.

$M_{EMAIL}\left(CLIENTE \bowtie \left(\rho_{DATARITIRO \leftarrow DATARITIRO1}\left(\delta_{DATARITIRO = DATARITIRO1}\left(PRENOTAZIONE \bowtie PRENOTAZIONE\right)\right)\right) \bowtie AGENZIA\right)$

Suggerimento per verifica/autovalutazione: Per ogni interrogazione, dopo averla formulata, effettuare i controlli richiesti e validare con V se si ritiene che il controllo sia superato, con X se si ritiene che non lo sia.

Verifica/autovalutazione	*a)*	*b)*
L'interrogazione formulata è corretta dal punto di vista dei vincoli di schema		
La richiesta e l'interrogazione formulata restituiscono una relazione con lo stesso schema		
La richiesta e l'interrogazione formulata sono entrambe monotone/non monotone		

COGNOME	NOME	MATRICOLA

Su una piccola istanza, la richiesta e l'interrogazione formulata restituiscono lo stesso risultato

Esercizio II.B - SQL

In riferimento al seguente schema relazionale (semplificazione dello schema dell'esercizio I)

AGENZIA (NomeAg, Indirizzo, Tel, Città, Stato)
VEICOLO (Targa, CategoriaV, Modello, Marca, Alimentazione, DataImm)
CLIENTE (IdCli, Nome, Indirizzo, Tel, *Email*, DataN, *NumeroPat*, ScadenzaPat)
PRENOTAZIONE (IdP, *Targa*$^{\text{VEICOLO}}$, *DataRitiro*, DataConsegna, AgRitiro$^{\text{AGENZIA}}$, AgConsegna$^{\text{AGENZIA}}$, IdCli$^{\text{LIENTE}}$, Stato, Importo)
NOLEGGIO (IdP$^{\text{PRENOTAZIONE}}$, KmIniziali, KmFinali, Carburante, Costo)
GUIDATORE_AGG (IdP$^{\text{PRENOTAZIONE}}$, IdCli$^{\text{CLIENTE}}$)

Formulare le seguenti interrogazioni in **SQL**.

1. Determinare le targhe dei veicoli elettrici che non sono mai stati noleggiati in un'agenzia e riconsegnati in un'altra.

```
SELECT DISTINCT TARGA
FROM VEICOLO
WHERE MODELLO = 'ELETTRICO' AND NOT IN ( SELECT DISTINCT TARGA
                                         FROM PRENOTAZIONE
                                         WHERE AGRITIRO != AGCONSEGNA )
```

2. Determinare i dati di contatto (email e telefono) del cliente che ha effettuato il noleggio (concluso) di durata massima.

```
SELECT C.EMAIL, C.TEL
FROM CLIENTE C
```

15420765101 11454232046 27401706422 75721021601 14307131652 43135317123 234115447451

COGNOME	NOME	MATRICOLA

Op.	Funzionalità	Cond.	Semantica
Π_A	$\mathcal{R}(U) \rightarrow \mathcal{R}(A)$	$A \subseteq U$	$\Pi_A(R) = \{t[A] \mid t \in R\}$
σ_F	$\mathcal{R}(U) \rightarrow \mathcal{R}(U)$	$A(F) \subseteq U$	$\sigma_F(R) = \{t \mid t \in R \wedge F(t)\}$
$\times$	$\mathcal{R}(U) \times \mathcal{R}(V) \rightarrow \mathcal{R}(U \cup V)$	$U \cap V = \emptyset$	$R_1 \times R_2 = \{t_1 \cdot t_2 \mid t_1 \in R_1 \wedge t_2 \in R_2\}$
$\cup$	$\mathcal{R}(U) \times \mathcal{R}(U) \rightarrow \mathcal{R}(U)$		$R_1 \cup R_2 = \{t \mid t \in R_1 \vee t \in R_2\}$
$-$	$\mathcal{R}(U) \times \mathcal{R}(U) \rightarrow \mathcal{R}(U)$		$R_1 - R_2 = \{t \mid t \in R_1 \wedge t \notin R_2\}$
$\cap$	$\mathcal{R}(U) \times \mathcal{R}(U) \rightarrow \mathcal{R}(U)$		$R_1 \cap R_2 = \{t \mid t \in R_1 \wedge t \in R_2\}$
$\bowtie_F$	$\mathcal{R}(U) \times \mathcal{R}(V) \rightarrow \mathcal{R}(U \cup V)$	$U \cap V = \emptyset$	$R_1 \bowtie_F R_2 = \{t_1 \cdot t_2 \mid t_1 \in R_1 \wedge t_2 \in R_2 \wedge F(t_1 \cdot t_2)\}$
$\bowtie$	$\mathcal{R}(U) \times \mathcal{R}(V) \rightarrow \mathcal{R}(U \cup V)$		$R_1 \bowtie R_2 = \{t \mid t[U] \in R_1 \wedge t[V] \in R_2\}$
$\div$	$\mathcal{R}(U) \times \mathcal{R}(V) \rightarrow \mathcal{R}(U \setminus V)$	$V \subset U$	$R_1 \div R_2 = \{t \mid \forall t_2 \in R_2 \, \exists t_1 \in R_1 \; t.c. \; t_1[U \setminus V] = t, t_1[V] = t_2\}$

$\rho_{A \leftarrow A'}$ $R(\mathrm{U}) \rightarrow R(\mathrm{U} \backslash \mathrm{A} \cup \mathrm{A}')$ $\mathrm{A} \subseteq \mathrm{U}$

SQL

SELECT *column_name(s)* **FROM** *table_name*	Select data from a table.
SELECT * FROM *table_name*	Select all data from a table.
SELECT DISTINCT *column_name(s)* **FROM** *table_name*	Select only distinct (different) data from a table.
SELECT *column_name(s)* **FROM** *table_name* **WHERE** *column operator value* **AND** *column operator value* **OR** *column operator value* **AND (... OR ...)** **...**	Select only certain data from a table.

Operato

Operator	
=	Equal
<>	Not equal
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
BETWEEN	Between an inclusive range
LIKE	Search for a pattern. A "%" sign can be used to define wildcards (missing letters in the pattern)

SELECT *column_name(s)* **FROM** *table_name* **WHERE** *column_name* **IN (***value1***,** *value2***, ...)**	The IN operator may be used if you know the exact value you want to return for at least one of the columns.
SELECT *column_name(s)* **FROM** *table_name* **ORDER BY** *row_1***,** *row_2* **DESC,** *row_3* **ASC, ...**	Select data from a table with sort the rows. **ASC** (ascend) is a alphabetical and numerical order (optional) **DESC** (descend) is a reverse alphabetical and numerical order

SELECT *column_1***, ...,** ***AGGREGATE_FUN*****(***agg_column_name***)** **FROM** *table_name* **GROUP BY** *group_column_name*	Without the GROUP BY all the tuples in the table are gouped together

ome aggregate

Function	
AVG(column)	Returns the average value of a column
COUNT(column)	Returns the number of rows (without a NULL value) of a column
MAX(column)	Returns the highest value of a column
MIN(column)	Returns the lowest value of a column
SUM(column)	Returns the total sum of a column

SELECT *column_1***, ...,** ***AGGREGATE_FUN*****(***agg_column_name***)** **FROM** *table_name* **GROUP BY** *group_column_name* **HAVING** **SUM(***group_column_name***)** *condition value*	HAVING... was added to SQL because the WHERE keyword could not be used against aggregate functions (like SUM), and without HAVING... it would be impossible to test for result conditions.
SELECT *column_name* **AS** *column_alias* **FROM** *table_name*	Column name alias
SELECT *table_alias.column_name* **FROM** *table_name* **AS** *table_alias*	Table name alias
SELECT *column_1_name***,** *column_2_name***, ...** **FROM** *first_table_name* **JOIN** *second_table_name* **ON** *first_table_name.keyfield* **=** *second_table_name.foreign_keyfield*	The (INNER) JOIN returns all rows from both tables where there is a match. If there are rows in first table that do not have matches in second table, those rows will not be listed. If this is not the intended behavior, use OUTER JOIN instead (RIGHT/LEFT/FULL)
SQL_Statement_1 **UNION** *SQL_Statement_2*	Select all different values from *SQL_Statement_1* and *SQL_Statement_2*

COGNOME	NOME	MATRICOLA

PARTE III. DOMANDE, SOLO PER 12 CFU

a) Discutere come le tecniche per la gestione dei file possono influenzare le prestazioni di un DBMS.

b) Descrivere la struttura di un indice ad albero e il costo per eseguire una operazione di selezione, con condizione di uguaglianza.

c) Definire e illustrare con un esempio il concetto di piano di esecuzione fisico.