COMP2411 Assignment

Name: Wang Yuqi
Student ID: [redacted]

Question A

Staff Table

```
CREATE DOMAIN STAFF_EMAIL AS VARCHAR(50)
CHECK (VALUE LIKE "%@staff.comp.polyu.hk");
CREATE DOMAIN STAFF_POS AS VARCHAR(20)
CHECK (VALUE IN ("Assistant Professor", "Associate Professor", "Professor");
CREATE DOMAIN OFFICE AS CHAR(5)
CHECK (VALUE LIKE "PQ___");
CREATE DOMAIN SALARY AS INTEGER
CHECK (VALUE BETWEEN 80000 AND 160000);

CREATE TABLE Staff(
  ID INTEGER NOT NULL,
  Name VARCHAR(50) NOT NULL,
  Email STAFF_EMAIL NOT NULL,
  Position STAFF_POS NOT NULL,
  Office OFFICE NOT NULL,
  Salary SALARY NOT NULL,
  PRIMARY KEY(ID)
);
```

Student Table

```
CREATE DOMAIN STU_EMAIL AS VARCHAR(50)
CHECK (VALUE LIKE "%@student.comp.polyu.hk");
CREATE DOMAIN STU_TYPE AS VARCHAR(20)
CHECK (VALUE IN ("Domestic", "International")):
CREATE DOMAIN STU_MAJOR AS CHAR(2)
CHECK (VALUE IN ("CS", "IT", "DS", "AI"));
CREATE DOMAIN GPA AS DECIMAL(2,1)
CHECK (VALUE BETWEEN 0 AND 4.3);

CREATE TABLE Student(
  ID INTEGER NOT NULL,
  Name VARCHAR(50) NOT NULL,
  Email STU_EMAIL NOT NULL,
  Type STU_TYPE NOT NULL,
  Major STU_MAJOR NOT NULL,
  AvgGrade GPA NOT NULL,
  Advisor INTEGER NOT NULL,
  PRIMARY KEY (ID),
  FOREIGN KEY (Advisor) REFERENCES Staff(ID) ON DELETE RESTRICT
);
```

Course Table

```
CREATE DOMAIN CREDIT AS INTEGER
CHECK (VALUE BETWEEN 1 AND 6):

CREATE TABLE Course (
  ID INTEGER NOT NULL,
  Name VARCHAR(50) NOT NULL,
  Credits CREDIT NOT NULL,
  PRIMARY KEY(ID)
):
```

Teach Table

```
CREATE DOMAIN SEM AS CHAR(6)
CHECK (VALUE IN ("Spring", "Summer", "Fall"));
CREATE DOMAIN EVAL AS DECIMAL(2,1)
CHECK (VALUE BETWEEN 0 AND 4);

CREATE TABLE Teach(
  StaffID INTEGER NOT NULL,
  CourseID INTEGER NOT NULL,
  Semester SEM NOT NULL,
  Evaluation EVAL NOT NULL
  PRIMARY KEY (StaffID, CourseID, Semester),
  FOREIGN KEY CourseID REFERENCES Course(ID) ON DELETE CASCADE,
  FOREIGN KEY StaffID REFERENCES Staff(ID) ON DELETE CASCADE
);
```

Enrolment

```
CREATE TABLE Enrolment(
  StudentID INTEGER NOT NULL,
  CourseID INTEGER NOT NULL,
  Grade GPA NOT NULL,
  PRIMARY KEY (StudentID, CourseID),
  FOREIGN KEY StudentID REFERENCES Student(ID) ON DELETE CASCADE,
  FOREIGN KEY CourseID REFERENCES Course(ID) ON DELETE RESTRICT
);
```

Question B

B(1)

We are given $R(A, B, C, D, E, F)$, and functional dependency (FD):

$$
\begin{aligned}
B &\to D \\
E &\to BCF \\
AB &\to C \\
AC &\to B \\
AD &\to E \\
BC &\to A
\end{aligned}
$$

Closure of B:

$$\begin{aligned} B &\to D \text{ (cannot expand further)} \\ B^+ &= \{B, D\} \end{aligned}$$

Closure of E:

$$\begin{aligned} E &\to BCF \\ &\to ABCF \ (\because BC \to A) \\ &\to ABCDF \ (\because B \to D) \\ E^+ &= \{A, B, C, D, E, F\} \end{aligned}$$

Closure of AB:

$$\begin{aligned} AB &\to C \\ &\to CD \ (\because B \to D) \\ &\to CDE \ (\because AD \to E) \\ &\to CDEF \ (\because E \to BCF) \\ (AB)^+ &= \{A, B, C, D, E, F\} \end{aligned}$$

Closure of AC:

$$\begin{aligned} AC &\to B \\ &\to BD \ (\because B \to D) \\ &\to BDE \ (\because AD \to E) \\ &\to BDEF \ (\because E \to BCF) \\ (AC)^+ &= \{A, B, C, D, E, F\} \end{aligned}$$

Closure of AD:

$$\begin{aligned} AD &\to E \\ &\to BCEF \ (\because E \to BCF) \\ (AD)^+ &= \{A, B, C, D, E, F\} \end{aligned}$$

Closure of BC:

$$\begin{aligned} BC &\to A \\ &\to AD \ (\because B \to D) \\ &\to ADE \ (\because AD \to E) \\ &\to ADEF \ (\because E \to BCF) \\ (BC)^+ &= \{A, B, C, D, E, F\} \end{aligned}$$

Therefore the candidate keys for R are: $\{E\}, \{AB\}, \{AC\}, \{A, D\}, \{B, C\}$

B(2)
Closure of A, C, G, CE, BG:

$$\begin{aligned} A^+ &= \{A, C, D, E, F\} \\ C^+ &= \{C, E, F\} \\ G^+ &= \{A, C, D, E, F, G\} \\ (CE)^+ &= \{C, E, F\} \\ (BG)^+ &= \{A, B, C, D, E, F, G, H\} \end{aligned}$$

Therefore, the candidate key of $R(A, B, C, D, E, F, G, H)$ is $\boldsymbol{\{B, G\}}$

2NF: no partial dependency
$G \to A$ violates 2NF as it is a partial dependency on the candidate key $\{B, G\}$. Therefore, we need to decompose R into R_1, R_2, where, the candidate key of $\{R_1\}$ is still $\{B, G\}$, and the candidate key of R_2 is $\{G\}$.

However, since $A \to CDEF$, it is better to keep $\{C, D, E, F\}$ together with $\{A\}$.
Otherwise, we would be implicitly breaking the transitive dependency $BG \to A \to CDEF$ during the decomposition, prematurely performing 3NF.

Hence, the 2NF decomposition is more appropriately written as:

$$R\big(\underline{B, G}, A, C, D, E, F, H\big) \xrightarrow{\text{decompose}} R_1\big(\underline{B, G}, H\big),\ R_2(\underline{G}, A, C, D, E, F)$$

3NF: no transitive dependency
$C \to EF$ violates 3NF since $G \to C \to EF$.
$CE \to F$ similarily violates 3NF since $G \to CE \to F$
$A \to CD$ also violates 3NF since $G \to A \to CD$
Therefore, we need to decompose R_2 further R_3, R_4, R_5:

$$R_2(\underline{G}, A, C, D, E, F) \xrightarrow{\text{decompose}} R_3(\underline{G}, A),\ R_4(\underline{A}, C, D),\ R_5(\underline{C}, E, F)$$

BCNF: all determinants are candidate keys
No need to decompose further. Already satisfies BCNF.

Question C

C(1)
Linear Search:

$$\left\lfloor \frac{1000}{50} \right\rfloor = 20 \text{ records/block}$$
$$\left\lceil \frac{6400}{20} \right\rceil = 320 \text{ blocks}$$

$$\therefore 320 \text{ block access}$$

Binary Search:

$$\left\lceil \frac{6400 - 6000}{20} \right\rceil = 20 \text{ blocks (data retrieval)}$$
$$\lceil \log_2 320 \rceil = 9 \text{ blocks (accessed)}$$

$$\therefore 20 + 9 = 29 \text{ block access}$$

C(2)
Since VID is an integer (4 bytes), the index entry size is:

$$4 + 6 = 10 \text{ bytes}$$
$$\left\lfloor \frac{1000}{10} \right\rfloor = 100 \text{ index/block}$$

Since there are 320 data blocks, and VID > 6000 takes up the last 20 blocks *(from c.1)*,

$$\left\lceil \frac{320}{100} \right\rceil = 4 \text{ index blocks}$$
$$\lceil \log_2 4 \rceil = 2 \text{ index block access}$$

$$\therefore 2 + 20 = 22 \text{ block access}$$

C(3)
Sorted:
Since each block can have a maxium of 100 indices *(from c.2)*, then the average number of index at each node is:

$$100 \times 0.6 = 60 \text{ index/block}$$
$$\lceil \log_{60} 6400 \rceil = 3 \text{ levels (tree height)}$$

Since there's a total of 20 data blocks (if sorted) *(from c.1*, we simply need to find the 1st index of these 20 data blocks and retrieve from there on.

$$\therefore 3 + 20 = 23 \text{ block access}$$

Unsorted:
Since the VID is unordered, we would need to retrieve every crecord with VID > 6000 in the B^+ tree and utilize **secondary indexing**. Therefore, the number of access within the B^+ tree is:

$$3 + \left\lceil \frac{400}{60} \right\rceil + 400 = 410 \text{ access}$$

C(4)
Since there are 6400 rows, each column takes up

$$\frac{6400}{8} = 800 \text{ bytes}$$
$$\left\lceil \frac{800}{1000} \right\rceil = 1 \text{ block}$$

Then, *Maker* takes up: $1 \times 80 = 80$ blocks
Color attribute: $1 \times 10 = 10$ blocks

Therefore, **answer 1**:

$$80 + 10 = 90 \text{ blocks}$$

—

Fetching the two bitmap columns where Maker = “Toyota” and Color = “Black” takes 2 access attemts (1 + 1 = 2). By doing so, we obtain the 40 rows that satisfies the conditions. Afterwards, the 40 rows can be retrieved in 40 cost.

Therefore, **answer 2**:

$$40 + 2 = 42 \text{ cost}$$