LAB8

DBMS

Group Members:

Jay Patel(201601130)

Shivam Bhavsar(201601414)

Utsav Rajpara(201601426)

QUESTION 1

Minimal FD set:

- {TN,DAY} -> {SRC_SCODE,DST_SCODE}
- {TN,SCODE,DATE} -> {EAT,SAT,SDT}

BCNF Decomposition:

- R(TN,DAY,SRC_SCODE,DST_SCODE)
- R(TN,SCODE,DATE,EAT,SAT,SDT)

QUESTION 2

AVEREAGE LATE TIME FUNCTION:

```
CREATE OR REPLACE FUNCTION AVG_LATE()

RETURNS SETOF AVERAGE AS $AVG_LATE$

DECLARE

SUM INTEGER; RECORD AVERAGE; TOTAL INTEGER; AVG_TIME NUMERIC; SEC
Schedule%ROWTYPE;

RUN Runlog%ROWTYPE;

BEGIN

FOR SEC IN SELECT * FROM Schedule LOOP

        SUM :=0 ; TOTAL :=0;

        IF SEC.Scheduled_Arrival_Time IS NOT NULL THEN

                FOR RUN IN SELECT* FROM Runlog WHERE
Train_Number=SEC.Train_Number AND
        Station_code=SEC.Station_code LOOP

                        IF RUN.Actual_Arrival_Time IS NOT NULL THEN

                                IF RUN.Actual_Arrival_Time
>SEC.Scheduled_Arrival_Time THEN

                                SUM := SUM + RUN.Actual_Arrival_Time -
SEC.Scheduled_Arrival_Time;

                                TOTAL:=TOTAL+1;

                                END IF;

                        END IF;
```

```
            END LOOP;
        RECORD.Train_Number: = SEC.Train_Number;
        RECORD.Station_Code:=SEC.Station_Code;
        IF TOTAL>0 THEN
            AVG_TIME := SUM/TOTAL;
        ELSE
            AVG_TIME := 0;
        END IF;
        RECORD.AVG_TIME :=AVG_TIME; RETURN NEXT RECORD;
        END IF;
END LOOP;
RETURN;
END $AVG_LATE$ LANGUAGE    'plpgsql';
```

QUESTION 3

Trigger function:

```
CREATE OR REPLACE FUNCTION sales_change()

RETURNS TRIGGER AS $$

BEGIN

IF TG_OP = 'INSERT' THEN

        UPDATE Item SET Stock=Stock+NEW.Qty WHERE Code=NEW.ItemCode;

        RETURN NEW;

ELSIF TG_OP='UPDATE' THEN

        UPDATE Item SET Stock=Stock+NEW.Qty-OLD.Qty WHERE
Code=NEW.ItemCode;

        RETURN NEW;

ELSIF TG_OP='DELETE' THEN

        UPDATE Item SET Stock=Stock-OLD.Qty WHERE Code=NEW.ItemCode;

        RETURN NEW;

END IF;

END $$ LANGUAGE    'plpgsql';
```

Create Trigger:

CREATE TRIGGER **set_sales**

AFTER **INSERT** OR **UPDATE** OR **DELETE**

ON **SalesDetails**

FOR EACH ROW

EXECUTE PROCEDURE **sales_change**();