Application Development II

Paul Alger & David Goff

12/6/22

Paul's time: 6 hours

David's time: 9 hours

"We certify that all the work in this application development project is the complete work of Paul and David."

Signed:

Paul Alger
David Goff

Features implemented:

- All features implemented
  - David Goff
    - Created all html files and web page specifics
    - Created songusageview
    - Additions to create_service procedure
    - Worked on python methods
  - Paul Alger
    - Created all views (except song usage view)
    - Created add_song stored procedure
    - Created create_service stored procedure
    - Worked on python methods
- Bonus work: song adding feature implemented
- No known bugs

Link to screen recording:

Link to app dev 2 demonstration

if the link above does not work here is the raw link:

https://youtu.be/6lCSuir7MIs

Store Procedure Create Statements:

create_service

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `create_service`(IN currentServiceID INT,
IN newDateTime DATETIME, IN newTheme VARCHAR(50), IN newSongleader VARCHAR(50), OUT
success INT)
BEGIN
	DECLARE next_id INTEGER;
    DECLARE personid INTEGER;

SELECT
    person.Person_ID
INTO personid FROM
```

```
    person
WHERE
    CONCAT(person.First_Name, ' ', person.Last_Name) = newSongleader;
      SELECT
    MAX(Service_ID) + 1
INTO next_id FROM
    service;

    # Check service time
    IF newDateTime IN (SELECT Svc_DateTime FROM service) THEN
              SET success = 0; # Error
    ELSE
              INSERT INTO service (Service_ID, Svc_DateTime, Theme_Event)
              VALUES (next_id, newDateTime, newTheme);

        IF personid IS NOT NULL THEN
                  INSERT INTO fills_role (Service_ID, Person_ID, Role_Type,
Confirmed)
                  VALUES (next_id, personid, 'S', 'Y');
              END IF;

        INSERT INTO service_item (Service_ID, Seq_Num, Event_Type_ID, Title, Notes,
Confirmed, Person_ID, Ensemble_ID, Song_ID)
        SELECT next_id, service_item.Seq_num, service_item.Event_Type_ID,
service_item.Title, service_item.Notes, 'Y', null, null, null FROM service_item
        WHERE Service_id = currentServiceID;


        SET success = 1; # Success
    END IF;
END
```

add_song

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `add_song`(IN item_id INT, IN
song_name VARCHAR(50))
BEGIN
      DECLARE new_song_id INTEGER;
SELECT
    Song_ID
INTO new_song_id FROM
    song
WHERE
    Title = song_name;

UPDATE service_item
SET
    Song_ID = new_song_id
WHERE
    Service_Item_ID = item_id;

END
```

Service_view

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
```

```
    SQL SECURITY DEFINER
VIEW `service_view` AS
    SELECT
        `service`.`Service_ID` AS `service_ID`,
        `service`.`Svc_DateTime` AS `Svc_DateTime`,
        `service`.`Theme_Event` AS `Theme_Event`,
        `songleader_view`.`songleader_name` AS `songleader`,
        `organist_view`.`organist_name` AS `organist`,
        `pianist_view`.`pianist_name` AS `pianist`,
        `service_item`.`Seq_Num` AS `Seq_Num`,
        `event_type`.`Description` AS `event`,
        (CASE
            WHEN
                (`service_item`.`Person_ID` IS NOT NULL)
            THEN
                CONCAT(`person`.`First_Name`,
                        ' ',
                        `person`.`Last_Name`)
            WHEN (`service_item`.`Ensemble_ID` IS NOT NULL) THEN `ensemble`.`Name`
            ELSE NULL
        END) AS `name`,
        (CASE
            WHEN
                (`song`.`Song_Type` = 'H')
            THEN
                CONCAT(`song`.`Hymnbook_Num`,
                        ' - ',
                        `song`.`Title`)
            WHEN (`song`.`Song_Type` = 'C') THEN `song`.`Title`
            ELSE `service_item`.`Title`
        END) AS `Title`,
        `service_item`.`Notes` AS `notes`
    FROM
        (((((((`service_item`
        JOIN `service` ON ((`service_item`.`Service_ID` = `service`.`Service_ID`)))
        JOIN `event_type` ON ((`service_item`.`Event_Type_ID` =
`event_type`.`Event_Type_ID`)))
        LEFT JOIN `songleader_view` ON ((`service_item`.`Service_ID` =
`songleader_view`.`service_id`)))
        LEFT JOIN `organist_view` ON ((`service_item`.`Service_ID` =
`organist_view`.`service_id`)))
        LEFT JOIN `pianist_view` ON ((`service_item`.`Service_ID` =
`pianist_view`.`service_id`)))
        LEFT JOIN `person` ON ((`service_item`.`Person_ID` = `person`.`Person_ID`)))
        LEFT JOIN `ensemble` ON ((`service_item`.`Ensemble_ID` =
`ensemble`.`Ensemble_ID`)))
        LEFT JOIN `song` ON ((`service_item`.`Song_ID` = `song`.`Song_ID`)))
    ORDER BY `service_item`.`Seq_Num`
```

Songleader_view

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `songleader_view` AS
    SELECT
        `fills_role`.`Service_ID` AS `service_id`,
        CONCAT(`person`.`First_Name`,
```

```
                ' ',
                `person`.`Last_Name`) AS `songleader_name`
    FROM
        (`person`
        JOIN `fills_role` ON ((`person`.`Person_ID` = `fills_role`.`Person_ID`)))
    WHERE
        (`fills_role`.`Role_Type` = 'S')
```

Organist_view

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `organist_view` AS
    SELECT
        `fills_role`.`Service_ID` AS `service_id`,
        CONCAT(`person`.`First_Name`,
                ' ',
                `person`.`Last_Name`) AS `organist_name`
    FROM
        (`person`
        JOIN `fills_role` ON ((`person`.`Person_ID` = `fills_role`.`Person_ID`)))
    WHERE
        (`fills_role`.`Role_Type` = 'O')
```

Pianist_view

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `organist_view` AS
    SELECT
        `fills_role`.`Service_ID` AS `service_id`,
        CONCAT(`person`.`First_Name`,
                ' ',
                `person`.`Last_Name`) AS `organist_name`
    FROM
        (`person`
        JOIN `fills_role` ON ((`person`.`Person_ID` = `fills_role`.`Person_ID`)))
    WHERE
        (`fills_role`.`Role_Type` = 'O')
```

songusageview

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `songusageview` AS
    SELECT
        `song`.`Song_ID` AS `Song_Id`,
        `song`.`Song_Type` AS `Song_Type`,
        `song`.`Title` AS `Title`,
        `song`.`Hymnbook_Num` AS `Hymnbook_Num`,
        `song`.`Arranger` AS `Arranger`,
        `service`.`Svc_DateTime` AS `LastUsedDate`
```

```
    FROM
        ((`song`
        LEFT JOIN `service_item` ON ((`song`.`Song_ID` = `service_item`.`Song_ID`)))
        LEFT JOIN `service` ON ((`service_item`.`Service_ID` =
`service`.`Service_ID`)))
    WHERE
        (`song`.`Song_Type` <> 'C')
    ORDER BY `service`.`Svc_DateTime` , `song`.`Title`
```