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](https://youtu.be/6lCSuir7MIs)

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`