



세계 종자 보관소를 위한 DB 구현

2023-2 산업정보관리론

Team 23

2020147049 김현동

2021147048 유효창



CONTENTS

- # 01 **Introduction**
- # 02 **Create Table / ER Model**
- # 03 **Insert Data**
- # 04 **Create Function**
- # 05 **Create View**
- # 06 **Create Table / ER Model**
- # 07 **Create Procedure**
- # 08 **Conclusion**

#01 INTRODUCTION

1) 종자의 안전한 보관과 체계적인 관리:



- Global Seed Vault는 전 세계의 유전자 은행에서 보관 중인 작물 다양성의 종자를 안전하고 무료로 장기 보존.
- 우리가 구현하는 DB는 이러한 종자 보존의 안전한 보관과 체계적인 관리를 지원하여, 세계 각국의 유전자 자원을 효율적으로 관리할 수 있게 만들 것이다.

2) 종자 보존 기록의 유지와 연구 자료로의 가치:

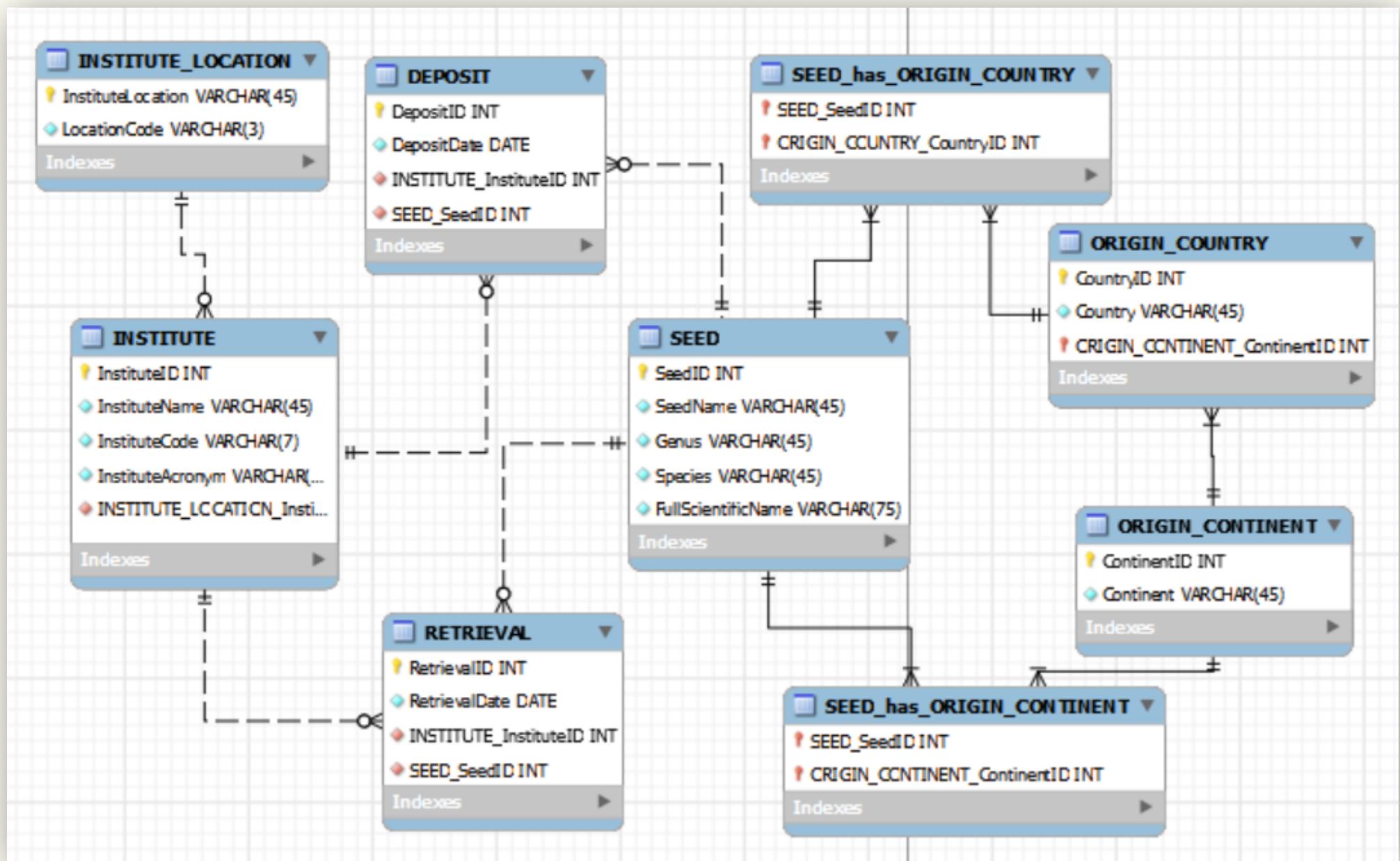


- Svalbard Global Seed Vault의 Seed Portal은 현재 100만 개 이상의 종자 샘플을 관리 중.
- 이러한 종자 보존 기록은 연구 및 미래 세대에게 유용한 자료가 될 것.
- 우리가 구현하는 DB는 이러한 보관 기록을 체계적으로 유지하고, 연구자들이 다양한 작물 종자의 다양성과 관련된 연구를 수행할 수 있는 토대로 제공될 것이다.

A screenshot of a website for the Svalbard Global Seed Vault. The header includes the logo 'SVALBARD GLOBAL SEED VAULT' and links for 'Contact', 'Login', 'Depositors', 'Depositor Guidelines', 'Search', 'About', and 'FAQ'. The main content area is titled 'Depositors' and shows statistics: '102 Total depositors registered' and '1,267,127 Total accessions registered'. A note states, 'This list includes depositors with seed samples shipped for safe storage at Svalbard Global Seed Vault.' Below this is a table with columns for 'Depositor ID', 'Name', 'Country', 'Number of accessions', and 'Last update'. A note at the bottom says, 'Click the column titles to sort the order of the list.'

Data Reference: <https://seedvault.nordgen.org/>

#02 CREATE TABLE - ER MODEL



- ER 모델을 제작한 뒤 이를 기반으로 CREATE TABLE을 하였다.
- Min/Max Cardinality는 오른쪽 ER 모델에서 확인할 수 있다.
- .sql 파일을 활용해 SQL Script로 CREATE TABLE을 할 수 있다.

#02 CREATE TABLE - INSTITUTE_LOCATION

- 종자를 예치하는 기관들의 소재지를 기입하는 테이블이다.
- InstituteLocation은 기관이 위치한 국가를 영문으로 표기한다.
- LocationCode는 기관이 위치한 국가를 세 개의 알파벳으로 표현한 코드이다. 예를 들어 France는 FRA, Republic of Korea는 KOR로 표기한다. 각 LocationCode는 unique하다.
- 새로운 기관이 등록될 때 기관의 소재지가 DB에 존재하지 않는다면 새롭게 등록을 해주어야 한다.

```
CREATE TABLE IF NOT EXISTS `mydb`.`INSTITUTE_LOCATION` (
  `InstituteLocation` VARCHAR(45) NOT NULL,
  `LocationCode` VARCHAR(3) NOT NULL,
  PRIMARY KEY (`InstituteLocation`),
  UNIQUE INDEX `LocationCode_UNIQUE` (`LocationCode` ASC) VISIBLE)
```

#02 CREATE TABLE - INSTITUTE

- 종자를 예치하는 기관들의 정보를 기입하는 테이블이다.
- InstituteID를 통해 각 기관들이 각기 다른 Row로 입력되게 했다. 기관이 입력되는 순으로 자동으로 부여된다.
- InstituteCode는 기관이 위치한 국가와 고유번호로 이루어진다. 알파벳 세 개와 3자리~4자리 숫자로 구성되며 프랑스에 위치한 National Institute for Agricultural Research의 경우 FRA040이 부여되었다. 정규표현식을 혼합해 설정 가능하지만 이 DB에서는 단순히 VARCHAR(7)로 설정했다.
- InstituteCOde와 InstituteName은 unique하다.
- InstituteAcronym은 기관 이름의 앞 글자를 따서 만든 코드로 Cherokee Nation의 경우 CN으로 표기된다.

```
CREATE TABLE IF NOT EXISTS `mydb`.`INSTITUTE` (
  `InstituteID` INT NOT NULL,
  `InstituteName` VARCHAR(45) NOT NULL,
  `InstituteCode` VARCHAR(7) NOT NULL,
  `InstituteAcronym` VARCHAR(45) NOT NULL,
  `INSTITUTE_LOCATION_InstituteLocation` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`InstituteID`),
  INDEX `fk_INSTITUTE_INSTITUTE_LOCATION1_idx` (`INSTITUTE_LOCATION_InstituteLocation` ASC) VISIBLE,
  UNIQUE INDEX `InstituteName_UNIQUE` (`InstituteName` ASC) VISIBLE,
  UNIQUE INDEX `InstituteCode_UNIQUE` (`InstituteCode` ASC) VISIBLE,
  CONSTRAINT `fk_INSTITUTE_INSTITUTE_LOCATION1`
    FOREIGN KEY (`INSTITUTE_LOCATION_InstituteLocation`)
    REFERENCES `mydb`.`INSTITUTE_LOCATION` (`InstituteLocation`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```

#02 CREATE TABLE - DEPOSIT

```
CREATE TABLE IF NOT EXISTS `mydb`.`DEPOSIT` (
  `DepositID` INT NOT NULL,
  `DepositDate` DATE NOT NULL,
  `INSTITUTE_InstituteID` INT NOT NULL,
  `SEED_SeedID` INT NOT NULL,
  PRIMARY KEY (`DepositID`),
  INDEX `fk_DEPOSIT_INSTITUTE1_idx` (`INSTITUTE_InstituteID` ASC) VISIBLE,
  INDEX `fk_DEPOSIT_SEED1_idx` (`SEED_SeedID` ASC) VISIBLE,
  CONSTRAINT `fk_DEPOSIT_INSTITUTE1`
    FOREIGN KEY (`INSTITUTE_InstituteID`)
    REFERENCES `mydb`.`INSTITUTE` (`InstituteID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_DEPOSIT_SEED1`
    FOREIGN KEY (`SEED_SeedID`)
    REFERENCES `mydb`.`SEED` (`SeedID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```

- 종자의 예치 기록을 담아 놓은 Table이다.
- DepositID는 종자가 예치된 순서에 따라 자동으로 부여된다.
- DepositDate에는 종자가 예치된 날짜가 기록된다.
- 한 번의 예치 당 한 종류의 종자만 예치가 가능하며 1회 예치 당 하나의 샘플이 예치된다고 가정한다.

#02 CREATE TABLE - RETRIEVAL

```
CREATE TABLE IF NOT EXISTS `mydb`.`RETRIEVAL` (
  `RetrievalID` INT NOT NULL,
  `RetrievalDate` DATE NOT NULL,
  `INSTITUTE_InstituteID` INT NOT NULL,
  `SEED_SeedID` INT NOT NULL,
  PRIMARY KEY (`RetrievalID`),
  INDEX `fk_RETRIEVAL_INSTITUTE1_idx` (`INSTITUTE_InstituteID` ASC) VISIBLE,
  INDEX `fk_RETRIEVAL_SEED1_idx` (`SEED_SeedID` ASC) VISIBLE,
  CONSTRAINT `fk_RETRIEVAL_INSTITUTE1`
    FOREIGN KEY (`INSTITUTE_InstituteID`)
    REFERENCES `mydb`.`INSTITUTE` (`InstituteID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_RETRIEVAL_SEED1`
    FOREIGN KEY (`SEED_SeedID`)
    REFERENCES `mydb`.`SEED` (`SeedID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```

- 종자의 반환 기록을 담아 놓은 Table이다.
- RetrievalID는 종자가 반환된 순서에 따라 자동으로 부여 된다.
- RetrievalDate에는 종자가 반환된 날짜가 기록된다.
- 한 번의 반환 당 한 종류의 종자만 반환이 가능하다고 가정 하며 한 번의 반환 당 하나의 샘플만 반환한다고 가정한다.

#02 CREATE TABLE - SEED

- 종자들의 정보를 기입하는 테이블이다. 아직 예치가 이루어지지 않았더라도 기재 가능하다.
- SeedID는 종자가 등록된 순서대로 자동으로 부여되는 숫자이다.
- SeedName은 종자의 이름(Alpine Strawberry, CommonWheat 등)을 포함한다. 각 종자별로 하나의 이름만이 존재한다고 가정한다.
- FullScientificName은 종자의 학명을 나타내며 unique한 값을 갖는다.

```
CREATE TABLE IF NOT EXISTS `mydb`.`SEED` (
  `SeedID` INT NOT NULL,
  `SeedName` VARCHAR(45) NOT NULL,
  `Genus` VARCHAR(45) NOT NULL,
  `Species` VARCHAR(45) NOT NULL,
  `FullScientificName` VARCHAR(75) NOT NULL,
  PRIMARY KEY (`SeedID`),
  UNIQUE INDEX `FullScientificName_UNIQUE` (`FullScientificName` ASC) VISIBLE)
```

#02 CREATE TABLE - ORIGIN_CONTINENT

- 종자를 수집한 대륙의 정보가 기입되는 Table이다.
- ContinentID는 대륙별로 임의로 지정되었다.
- Continent는 총 7개의 대륙으로 구성되었으며, Asia, Europe, Africa, North America, South America, Oceania, Antarctica가 등록되었다.

```
CREATE TABLE IF NOT EXISTS `mydb`.`ORIGIN_CONTINENT` (
  `ContinentID` INT NOT NULL,
  `Continent` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`ContinentID`),
  UNIQUE INDEX `Continent_UNIQUE` (`Continent` ASC) VISIBLE)
```

#02 CREATE TABLE - ORIGIN_COUNTRY

```
CREATE TABLE IF NOT EXISTS `mydb`.`ORIGIN_COUNTRY` (
  `CountryID` INT NOT NULL,
  `Country` VARCHAR(45) NOT NULL,
  `ORIGIN_CONTINENT_ContinentID` INT NOT NULL,
  PRIMARY KEY (`CountryID`, `ORIGIN_CONTINENT_ContinentID`),
  INDEX `fk_ORIGIN_COUNTRY_ORIGIN_CONTINENT1_idx` (`ORIGIN_CONTINENT_ContinentID` ASC) VISIBLE,
  UNIQUE INDEX `Country_UNIQUE` (`Country` ASC) VISIBLE,
  CONSTRAINT `fk_ORIGIN_COUNTRY_ORIGIN_CONTINENT1`
    FOREIGN KEY (`ORIGIN_CONTINENT_ContinentID`)
    REFERENCES `mydb`.`ORIGIN_CONTINENT` (`ContinentID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```

- 종자를 수집한 국가의 정보가 기입되는 Table이다.
- CountryID는 국가별로 임의로 지정되었다.
- Country에는 국가의 이름이 들어가며 unique한 값이다.

#02 CREATE TABLE - SEED_has_ORIGIN_CONTINENT/COUNTRY

```
CREATE TABLE IF NOT EXISTS `mydb`.`SEED_has_ORIGIN_CONTINENT` (
  `SEED_SeedID` INT NOT NULL,
  `ORIGIN_CONTINENT_ContinentID` INT NOT NULL,
  PRIMARY KEY (`SEED_SeedID`, `ORIGIN_CONTINENT_ContinentID`),
  INDEX `fk_SEED_has_ORIGIN_CONTINENT_ORIGIN_CONTINENT1_idx` (`ORIGIN_CONTINENT_ContinentID` ASC) VISIBLE,
  INDEX `fk_SEED_has_ORIGIN_CONTINENT_SEED1_idx` (`SEED_SeedID` ASC) VISIBLE,
  CONSTRAINT `fk_SEED_has_ORIGIN_CONTINENT_SEED1`
    FOREIGN KEY (`SEED_SeedID`)
    REFERENCES `mydb`.`SEED` (`SeedID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_SEED_has_ORIGIN_CONTINENT_ORIGIN_CONTINENT1`
    FOREIGN KEY (`ORIGIN_CONTINENT_ContinentID`)
    REFERENCES `mydb`.`ORIGIN_CONTINENT` (`ContinentID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```

```
CREATE TABLE IF NOT EXISTS `mydb`.`SEED_has_ORIGIN_COUNTRY` (
  `SEED_SeedID` INT NOT NULL,
  `ORIGIN_COUNTRY_CountryID` INT NOT NULL,
  PRIMARY KEY (`SEED_SeedID`, `ORIGIN_COUNTRY_CountryID`),
  INDEX `fk_SEED_has_ORIGIN_COUNTRY_ORIGIN_COUNTRY1_idx` (`ORIGIN_COUNTRY_CountryID` ASC) VISIBLE,
  INDEX `fk_SEED_has_ORIGIN_COUNTRY_SEED1_idx` (`SEED_SeedID` ASC) VISIBLE,
  CONSTRAINT `fk_SEED_has_ORIGIN_COUNTRY_SEED1`
    FOREIGN KEY (`SEED_SeedID`)
    REFERENCES `mydb`.`SEED` (`SeedID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_SEED_has_ORIGIN_COUNTRY_ORIGIN_COUNTRY1`
    FOREIGN KEY (`ORIGIN_COUNTRY_CountryID`)
    REFERENCES `mydb`.`ORIGIN_COUNTRY` (`CountryID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```

- SEED와 ORIGIN_COUNTRY, SEED와 ORIGIN_CONTINENT의 M:N 관계를 표현하기 위해 존재하는 Association Table들을 생성하였다.

#03 INSERT DATA

- 실제 Svalbard Global Seed Vault의 데이터를 참조하여 CSV 파일로 제작한 가상의 데이터를 활용 (각 Table별 TableName.csv 파일 제출)
- Excel Data Import Wizard를 이용해 데이터를 넣었지만, 이후 Data Export를 이용해 sql 형태로도 변환했다.

DEPOSIT(DepositID, DepositDate, INSTITUTE_InstituteID, SEED_SeedID)

	DepositID	DepositDate	INSTITUTE_InstituteID	SEED_SeedID
▶	3	2015-02-27	3	3
	4	2015-02-27	3	3
	5	2015-02-27	3	3
	6	2015-02-27	3	4
	7	2015-02-27	3	4
	8	2015-02-27	3	4
	9	2015-02-27	3	4
	10	2016-06-23	1	1
	11	2016-06-23	1	1
	12	2016-10-18	4	5
	13	2016-10-18	4	5
	14	2016-10-18	4	5
	15	2016-10-18	4	5
	16	2016-10-18	4	5
	17	2016-10-18	4	5
	18	2016-10-18	4	6

#03 INSERT DATA

INSTITUTE(InstituteID, InstituteName, InstituteCode, InstituteAcronym, *INSTITUTE_LOCATION_InstituteLocation*)

	InstituteID	InstituteName	InstituteCode	InstituteAcronym	INSTITUTE_LOCATION_InstituteLocation
▶	1	National Institute for Agricultural Research	FRA040	INRA	France
	3	Natural Resources Institute Finland	FIN027	Luke	Finland
	4	Temasek Life Sciences Laboratory Limited	SGP008	TLL	Singapore
	5	Cherokee Nation	USA1005	CN	United States
	6	Baedduaegan National Arboretum	KOR048	BDNA	Republic of Korea
	7	Botanical Garden, University of Bonn	DEU038	ABGBONN	Germany
*	NULL	NULL	NULL	NULL	NULL

INSTITUTION_LOCATION(InstitutionLocation, LocationCode)

	InstitutionLocation	LocationCode
▶	Finland	FIN
	France	FRA
	Germany	DEU
	Italia	ITA
	Republic of Korea	KOR
	Singapore	SGP
	United States	USA
*	NULL	NULL

ORIGIN_CONTINENT(ContinentID, Continent)

	ContinentID	Continent
▶	1	Asia
	2	Europe
	3	Africa
	4	North America
	5	South America
	6	Oceania
	7	Antarctica
*	NULL	NULL

#03 INSERT DATA

SEED(SeedID, SeedName, Genus, Species, FullScientificName)

	SeedID	SeedName	Genus	Species	FullScientificName
▶	1	Common Wheat	Triticum	Triticum aestivum	Triticum aestivum subsp. aestivum L.
	2	Maize	Zea	Zea mays	Zea mays L.
	3	Scots Pine	Pinus	Pinus sylvestris	Pinus sylvestris
	4	Norway spruce	Picea	Picea abies	Picea abies L.
	5	Indica Rice	Oryza	Oryza sativa	Oryza sativa subsp. indica
	6	Japonica Rice	Oryza	Oryza sativa	Oryza sativa Japonica
	7	Common Bean	Phaseolus	Phaseolus vulgaris	Phaseolus vulgaris
	8	Buttercup Squash	Cucurbita	Cucurbita maxima	Cucurbita maxima
	9	Korean Creeping Raspberry	Rubus	Rubus oldhamii	Rubus oldhamii Miq.
	10	Great Burnet	Sanguisorba	Sanguisorba officinalis	Sanguisorba officinalis L.
	11	Spikenard	Aralia	Aralia cordata	Aralia cordata var. continentalis (Kit...
	12	Korean Mint	Agastache	Agastache rugosa	Agastache rugosa (Fisch. & Mey.) K...
	13	Woundwort	Solidago	Solidago virgaurea	Solidago virgaurea subsp. gigantea ...
	14	Goat's Beard	Aruncus	Aruncus dioicus	Aruncus dioicus var. kamtschaticus ...
	15	Siberian Veronicastrum	Veronicastrum	Veronicastrum sibiricum	Veronicastrum sibiricum (L.) Pennell
	16	Aqing Chive	Allium	Allium senescens	Allium senescens L.

SEED_has_ORIGIN_CONTINENT(SEED_SeedID, ORIGIN_CONTINENT_ContinentID)

	SEED_SeedID	ORIGIN_CONTINENT_ContinentID
▶	5	1
	6	1
	9	1
	10	1
	11	1
	12	1
	13	1
	14	1
	15	1
	16	1
	17	1

#03 INSERT DATA

SEED_has_ORIGIN_COUNTRY(SEED_SeedID, ORIGIN_COUNTRY_CountryID)

	SEED_SeedID	ORIGIN_COUNTRY_CountryID
▶	1	1
	2	2
	3	3
	4	3
	5	4
	6	4
	2	5
	7	5
	8	5
	9	6
	10	6
	11	6
	12	6
	13	6
	14	6
	15	6

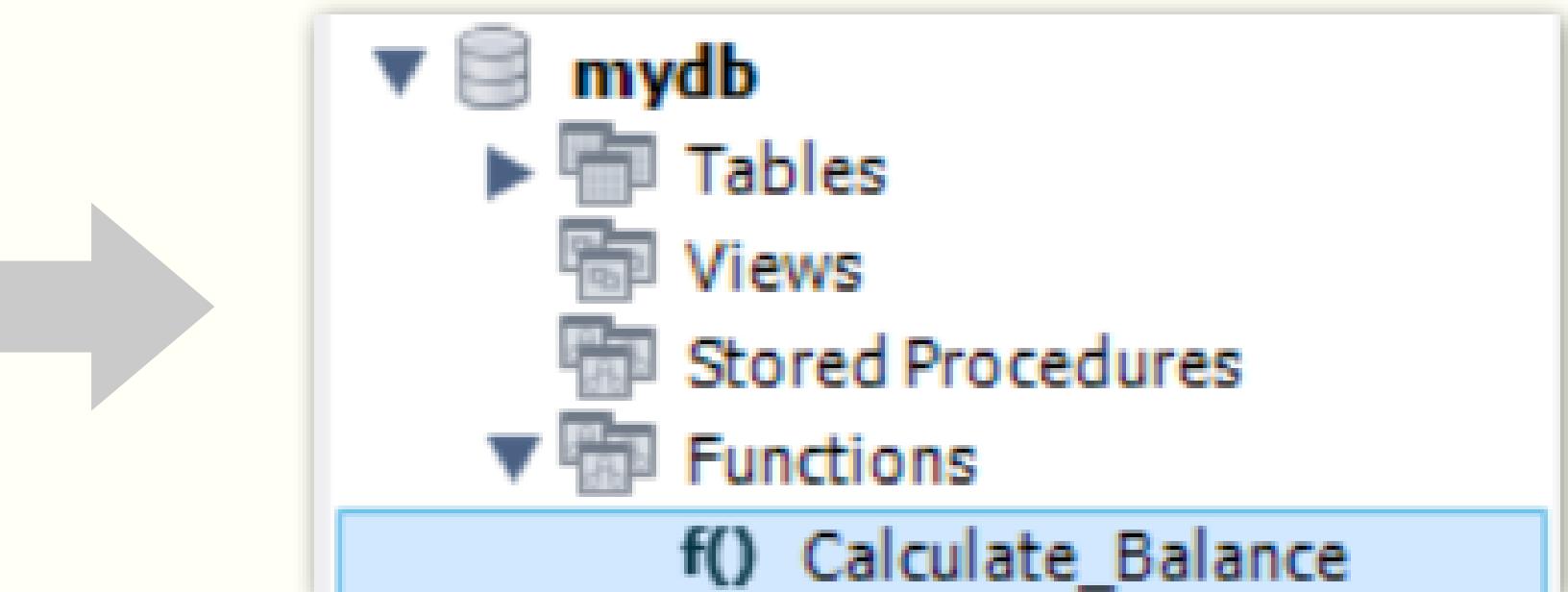
RETRIEVAL(RetrievalID, RetrievalDate, INSTITUTE_InstituteID, SEED_SeedID)

	RetrievalID	RetrievalDate	INSTITUTE_InstituteID	SEED_SeedID
*	NULL	NULL	NULL	NULL

#04 CREATE FUNCTION

- Calculate_Balance라는 Function을 만들어 종자 은행에 해당 종자의 샘플이 몇 개 남았는지 계산하는 함수이다. 샘플이 적을수록 수집의 필요성을 암시한다.
- 예치된 종자 샘플의 수에서 반환된 샘플의 수를 빼서 현재 종자 은행에 남은 해당 종자의 샘플 수를 계산한다.

```
/* Seed Balance Calculator */
DELIMITER //
CREATE FUNCTION Calculate_Balance
(
varSeedDepositNumber INT,
varSeedRetrievalNumber INT
)
RETURNS INT DETERMINISTIC
BEGIN
DECLARE varCalculate_Balance INT;
SET varCalculate_Balance = varSeedDepositNumber - varSeedRetrievalNumber;
RETURN varCalculate_Balance;
END
//
DELIMITER ;
```

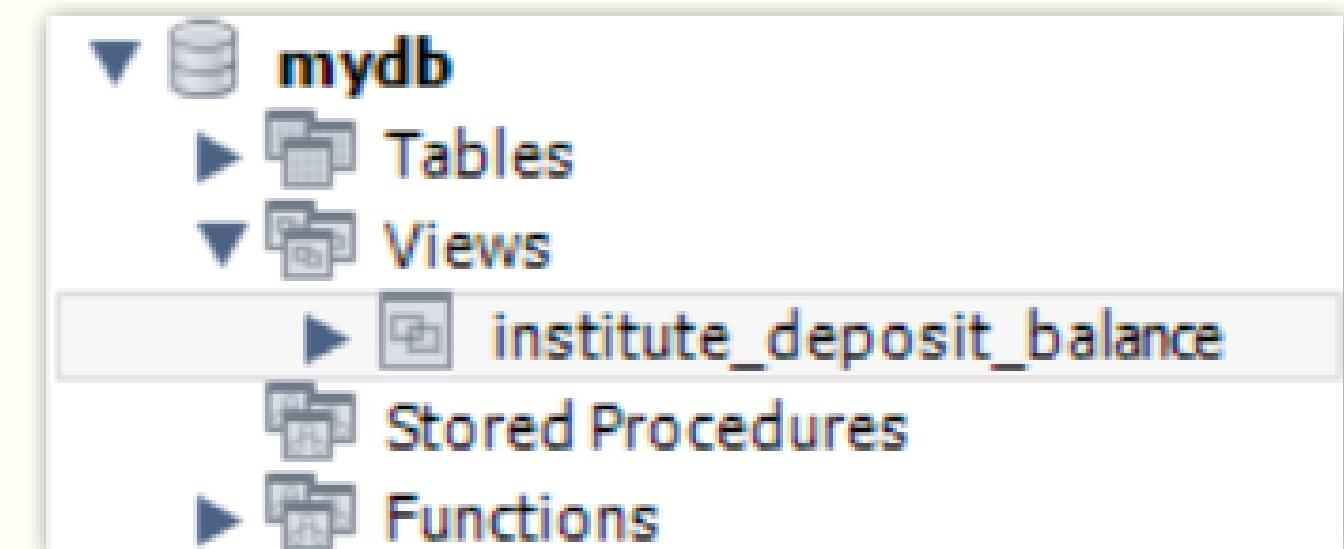
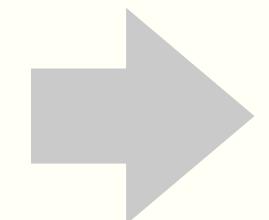


#05 CREATE VIEW

- INSTITUTE_DEPOSIT_BALANCE라는 View를 만들어 각 기관이 예치한 종자의 종별로 몇 개의 샘플을 예치하였는지 보여준다.
- 앞에서 만들었던 Calculate_Balance 함수를 활용했다.

```
/*Create View*/
CREATE VIEW INSTITUTE_DEPOSIT_BALANCE AS
SELECT I.InstituteCode AS InstituteCode,
       I.InstituteName AS Institute,
       Calculate_Balance(COUNT(D.DepositID), COALESCE(COUNT(R.RetrievalID), 0)) AS Balance,
       S.SeedName AS SeedName,
       S.Genus AS Genus,
       S.Species AS Species,
       S.FullScientificName AS FullScientificName
  FROM DEPOSIT D
 LEFT JOIN RETRIEVAL R ON R.SEED_SeedID = D.SEED_SeedID
 JOIN SEED S ON S.SeedID = D.SEED_SeedID
 JOIN INSTITUTE I ON I.InstituteID = D.INSTITUTE_InstituteID
 GROUP BY I.InstituteCode, I.InstituteName, S.SeedName, S.Genus, S.Species, S.FullScientificName;

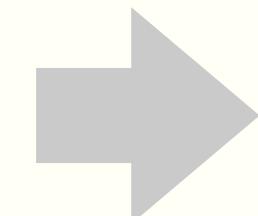
/*View Select By Institute, Order By Total Score DESC*/
SELECT *
  FROM institute_deposit_balance
 WHERE InstituteCode = 'DEU001'
 ORDER BY Balance DESC;
```



#05 CREATE VIEW

- VIEW에서는 ORDER BY를 사용할 수 없어, 추가로 ORDER BY Balance DESC;를 적용했다.
- 이를 통해 Balance가 높은 순, 즉 반환받을 수 있는 종자의 샘플 수가 많은 순서대로 예치한 종자의 정보를 확인할 수 있다.
- 기관의 코드를 활용해 기관의 코드, 이름과 함께 해당 기관에서 예치한 종자별로 샘플의 개수, 종자의 이름, 속, 종, 아종 까지 VIEW를 통해 확인 가능하게 만들었다.

```
SELECT *
FROM institute_deposit_balance
WHERE InstituteCode = 'DEU038'
ORDER BY Balance DESC;
```



	InstituteCode	Institute	Balance	SeedName	Genus	Species	FullScientificName
▶	DEU038	Botanical Garden, University of Bonn	3	Head Lettuce	Lacuta	Lactuca sativa	Lactuca sativa
	DEU038	Botanical Garden, University of Bonn	2	Wild Cabbage	Brassica	Brassica oleracea	Brassica oleracea
	DEU038	Botanical Garden, University of Bonn	2	Tomato	Solanum	Solanum lycopersicum	Solanum lycopersicum
	DEU038	Botanical Garden, University of Bonn	1	Cucumber	Cucumis	Cucumis sativus	Cucumis sativus
	DEU038	Botanical Garden, University of Bonn	1	Leaf Lettuce	Lacuta	Lactuca sativa	Lactuca sativa
	DEU038	Botanical Garden, University of Bonn	1	Spinach	Spinacia	Spinacia oleracea	Spinacia oleracea
	DEU038	Botanical Garden, University of Bonn	1	Italian Corn-salad	Valerianella	Valerianella eriocarpa	Valerianella eriocarpa
	DEU038	Botanical Garden, University of Bonn	1	Common Corn-salad	Valerianella	Valerianella locusta	Valerianella locusta

#06 CREATE TRIGGER

```
DELIMITER //

CREATE TRIGGER seed_insert_trigger
BEFORE INSERT ON SEED
FOR EACH ROW
BEGIN
    DECLARE species_name VARCHAR(45);
    DECLARE genus_name VARCHAR(45);
    DECLARE space_count INT;

    SET space_count = LENGTH(NEW.FullScientificName) - LENGTH(REPLACE(NEW.FullScientificName, ' ', ''));

    IF space_count >= 2 THEN
        SET genus_name = SUBSTRING_INDEX(NEW.FullScientificName, ' ', 1);
        SET species_name = SUBSTRING_INDEX(NEW.FullScientificName, ' ', 2);
    ELSEIF space_count = 1 THEN
        SET species_name = NEW.FullScientificName;
        SET genus_name = SUBSTRING_INDEX(NEW.FullScientificName, ' ', 1);
    END IF;

    IF NEW.Species IS NULL THEN
        SET NEW.Species = species_name;
    END IF;

    IF NEW.Genus IS NULL THEN
        SET NEW.Genus = genus_name;
    END IF;

    SET NEW.SeedName = 'Information Needed';

    IF NEW.SeedID IS NULL THEN
        SET NEW.SeedID = (SELECT COALESCE(MAX(SeedID), 0) + 1 FROM SEED);
    END IF;
END;
// 

DELIMITER ;
```

- 종자 은행 DB에 등록되지 않았던 새로운 종/아종의 종자가 등록될 때를 대비한 SEED_INSERT_TRIGGER를 제작했다.
- 종자 은행 DB에 등록되지 않았던 새로운 종/아종의 종자를 등록할 때 해당 종자의 학명을 입력하면 자동으로 해당 종자의 속과 종을 입력하도록 TRIGGER를 제작했다.
- 새롭게 추가된 종/아종의 이름은 우선 'Information Needed'로 등록되게 하여 추후 해당 종자가 명명될 경우 UPDATE가 가능하도록 설정했다.

#06 CREATE TRIGGER

- 기존에 등록되어 있지 않은 'Citrus sinensis'가 SEED Table에 등록되는 상황을 가정했다.
- SEED Table에 등록될 때 SeedID, Genus, Species가 모두 자동으로 입력되는 것을 확인할 수 있다.
- 추후 SeedName은 'Orange'로 UPDATE 가능하다.

```
INSERT INTO SEED (FullScientificName)
VALUES ('Citrus sinensis');
```

	SeedID	SeedName	Genus	Species	FullScientificName
	18	Toringo Crabapple	Malus	Malus sieboldii	Malus sieboldii (Regel) Rehder
	19	Wild Cabbage	Brassica	Brassica oleracea	Brassica oleracea convar. capitata ...
	20	Cucumber	Cucumis	Cucumis sativus	Cucumis sativus L.
	21	Head Lettuce	Lacuta	Lactuca sativa	Lactuca sativa var. capitata L.
	22	Leaf Lettuce	Lacuta	Lactuca sativa	Lactuca sativa var. crispa L.
	23	Tomato	Solanum	Solanum lycopersicum	Solanum lycopersicum L.
	24	Spinach	Spinacia	Spinacia oleracea	Spinacia oleracea L.
	25	Italian Corn-salad	Valerianella	Valerianella eriocarpa	Valerianella eriocarpa Desv.
	26	Common Corn-salad	Valerianella	Valerianella locusta	Valerianella locusta (L.) Laterr.
	27	European Wild Pear	Pyrus	Pyrus pyraster	Pyrus pyraster (L.) Burgsd.
	28	Crab Apple	Malus	Malus sylvestris	Malus sylvestris (L.) Mill.
	29	Wild Strawberry	Fragaria	Fragaria vesca	Fragaria vesca L.
	30	Alpine Strawberry	Fragaria	Fragaria vesca	Fragaria vesca f. semperflorens
*	31	Information Needed	Citrus	Citrus sinensis	Citrus sinensis

#07 CREATE PROCEDURE

- 종자의 학명을 입력했을 때 해당 종자를 예치한 기관들과 각 기관들이 예치한 종자 샘플의 수, 기관의 정보 등을 반환하는 VIEW_SPECIES HOLDER라는 PROCEDURE를 제작했다.

```
/*Create Procedure View_Species_Holder*/  
  
DELIMITER //  
  
CREATE PROCEDURE VIEW_SPECIES HOLDER  
(  
    IN newSpecies VARCHAR(45)  
)  
  
BEGIN  
  
    DECLARE varBalance INT;  
  
    SELECT Calculate_Balance(COALESCE(COUNT(D.DepositID), 0), COALESCE(COUNT(R.RetrievalID), 0)) INTO varBalance  
    FROM DEPOSIT AS D  
    LEFT JOIN RETRIEVAL AS R ON R.SEED_SeedID = D.SEED_SeedID  
    JOIN SEED AS S ON S.SeedID = D.SEED_SeedID  
    JOIN INSTITUTE AS I ON I.InstituteID = D.INSTITUTE_InstituteID  
    WHERE S.FullScientificName = newSpecies;  
    IF (varBalance = 0)  
    THEN  
        SELECT 'There are no institute who holds the species.'  
        AS Search_Result;  
  
    ELSE  
        SELECT I.InstituteID, I.InstituteName, I.InstituteCode, I.INSTITUTE_LOCATION_InstituteLocation AS InstituteLocation,  
        Count(*) AS NumOfSeeds  
        FROM INSTITUTE AS I  
        JOIN DEPOSIT AS D ON I.InstituteID = D.INSTITUTE_InstituteID  
        LEFT JOIN RETRIEVAL AS R ON R.SEED_SeedID = D.SEED_SeedID  
        JOIN SEED AS S ON S.SeedID = D.SEED_SeedID  
        WHERE S.FullScientificName = newSpecies  
        GROUP BY I.InstituteID, I.InstituteName, I.InstituteCode, InstituteLocation;  
    END IF;  
  
END;  
//
```

- 종자의 학명별로 예치한 기관이 존재하는 경우와 존재하지 않는 경우의 차이를 확인할 수 있다.
- 해당 종자를 예치한 기관이 없는 경우, 예치한 기관이 없다는 내용의 안내문이 나오도록 하였다.

CALL VIEW_SPECIES HOLDER('Pinus sylvestris');

	InstituteID	InstituteName	InstituteCode	InstituteLocation	NumOfSeeds
▶	3	Natural Resources Institute Finland	FIN027	Finland	3

CALL VIEW_SPECIES HOLDER('Cucurbita maxima');

	InstituteID	InstituteName	InstituteCode	InstituteLocation	NumOfSeeds
▶	5	Cherokee Nation	USA1005	United States	1

CALL VIEW_SPECIES HOLDER('Citrus sinensis');

	Search_Result
▶	There are no institute who holds the species.

#08 CONCLUSION

Limitation

- 데이터 Input에 있어서 더 많은 제약 조건을 포함했다면, 보다 더 강건하고 정교한 DB를 구축할 수 있었을 것이다.
- 모든 기관, 종자, 국가 등에 대한 정보를 담지 못해 DB가 현실과는 약간 동떨어져 있을 수 있다.
- 데이터 수집 및 INSERT에 있어 직접 정보를 수집해 .csv 파일을 제작한 뒤 데이터를 INSERT해줘야 하는 과정에 어려움이 있었다. 데이터 크롤링 등을 결합하면, 데이터 수집 및 INSERT 과정이 수월해질 것으로 기대된다.



Possible Improvement

- 테이블에 들어갈 Value의 Format에 제약조건을 걸게 함으로써, 새로운 Value를 입력할 때 더 정확한 Input을 받을 수 있게 만들 수 있을 것이다.
- 이번 프로젝트에서는 하나의 종자 당 이름이 하나인 것으로 가정했지만, 실제로는 나라 또는 지역별로 부르는 이름이 달라 이에 대한 고려가 필요하다.
- 종자별 생육 특징 또는 정보들을 추가로 담을 수 있다면, 일종의 종자 백과사전으로써의 역할도 수행할 수 있을 것으로 기대된다.

THANK YOU