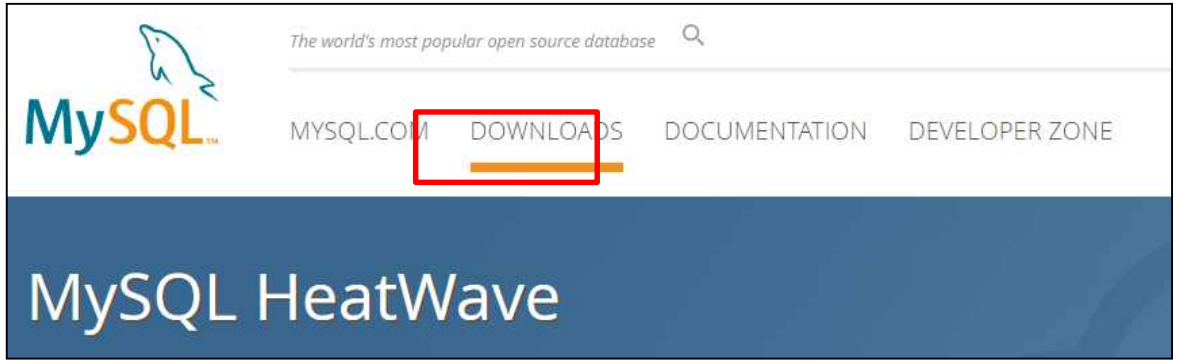
TABA_DB/SQL실습1

What is MySQL?

- MySQL은 가장 많이 사용되고 있는 관계형 데이터베이스 관리 시스템(RDBMS)
- MySQL은 오픈소스이며, 다중 사용자 및 다중 스레드 지원
- C, JAVA, PHP, R 등 다양한 프로그래밍 언어를 위한 API를 제공

- MySQL은 리눅스, 윈도우 등 다양한 운영체제에서 사용 가능하며, PHP와 웹 개발에 자주 사용(Apache, PHP, MySQL)

Install MySQL



Install MySQL

MySQL Community Downloads

- MySQL Yum Repository
- MySQL APT Repository
- MySQL SUSE Repository

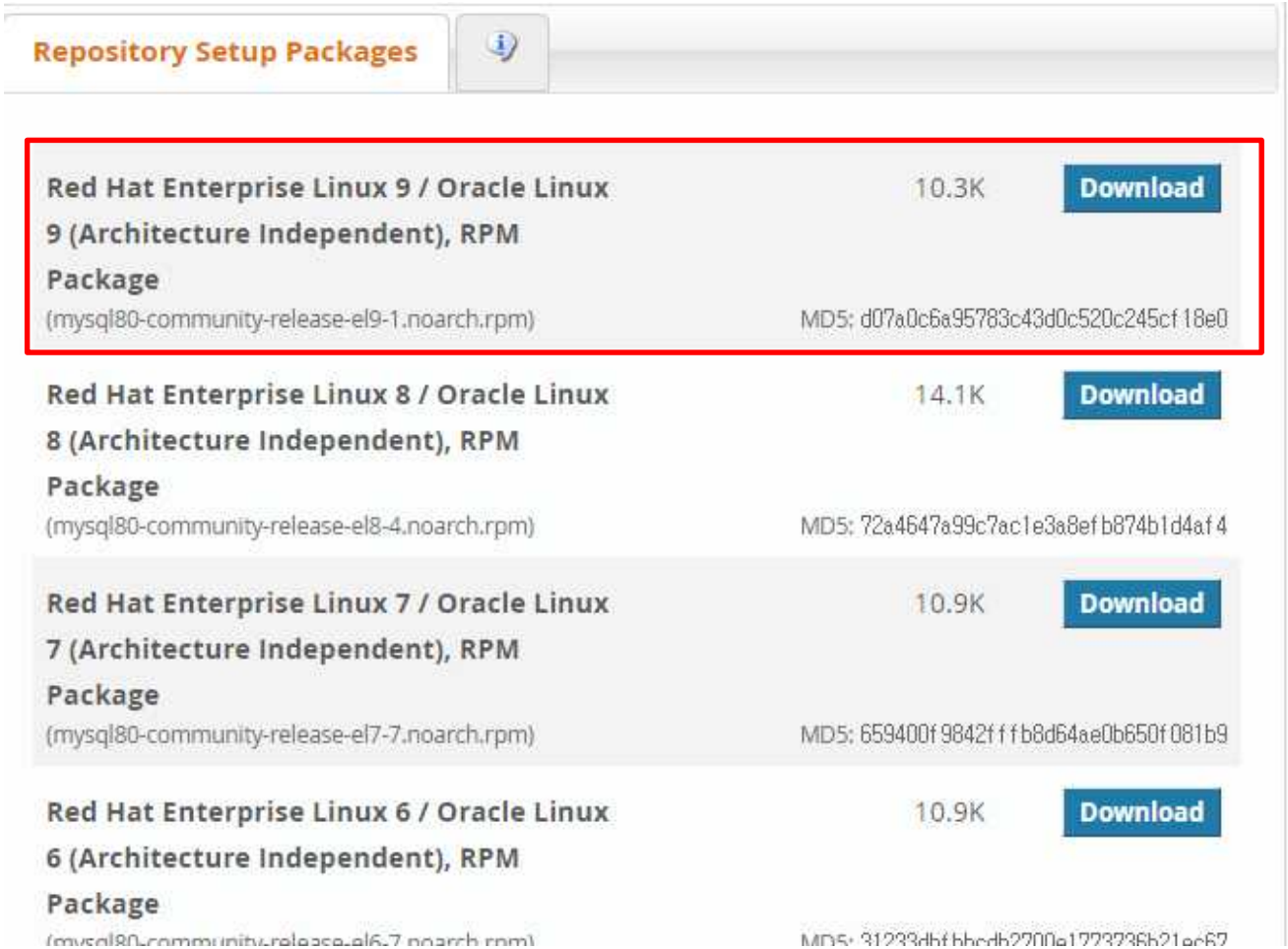
- MySQL Community Server
- MySQL Cluster
- MySQL Router
- MySQL Shell
- MySQL Operator
- MySQL Workbench

- MySQL Installer for Windows
- MySQL for Visual Studio

- C API (libmysqlclient)
- Connector/C++
- Connector/J
- Connector/NET
- Connector/Node.js
- Connector/ODBC
- Connector/Python
- MySQL Native Driver for PHP

- MySQL Benchmark Tool
- Time zone description tables
- Download Archives

Install MySQL



Install MySQL

No thanks, just start my download.

마우스 우측 클릭 - 링크 주소 복사

Install MySQL

- **MySQL repository** 설치

- sudo yum install -y https://dev.mysql.com/get/mysql84-community-release-el9-1.noarch.rpm

```
[root@localhost shjeon]# yum install -y https://dev.mysql.com/get/mysql80-community-release-el8-4.noarch.rpm
CentOS Stream 9 - BaseOS                                        112 kB/s | 5.9 MB     00:53
CentOS Stream 9 - AppStream                                     5.7 MB/s |  15 MB     00:02
CentOS Stream 9 - Extras packages                               6.5 kB/s | 8.5 kB     00:01
Last metadata expiration check: 0:00:01 ago on Wed 07 Sep 2022 11:04:58 AM KST.
mysql80-community-release-el8-4.noarch.rpm                       33 kB/s |  14 kB     00:00
Dependencies resolved.
================================================================================================
 Package                        Architecture      Version           Repository              Size
================================================================================================
Installing:
 mysql80-community-release      noarch            el8-4             @commandline            14 k

Transaction Summary
================================================================================================
Install  1 Package
```

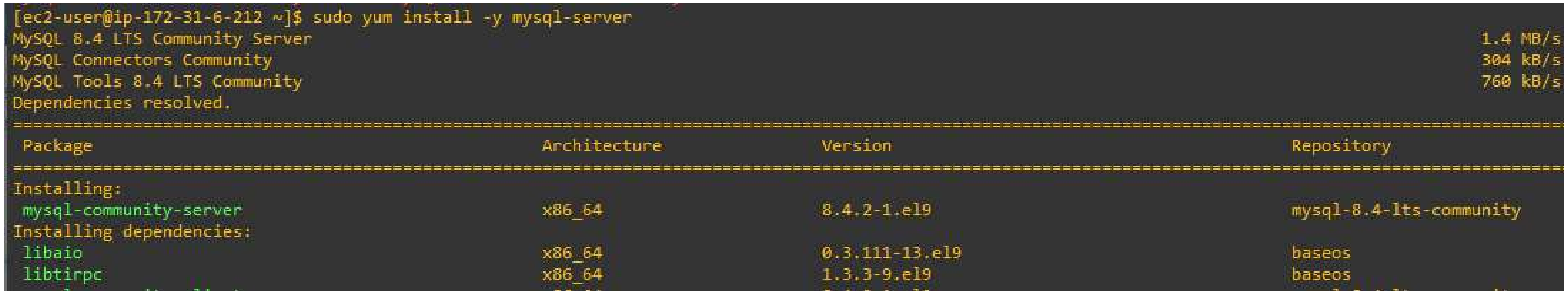
Install MySQL

- MySQL repository 확인
- yum repolist enabled | grep "mysql.*"

```
[ec2-user@ip-172-31-6-212 ~]$ yum repolist enabled | grep "mysql.*"
mysql-8.4-lts-community                 MySQL 8.4 LTS Community Server
mysql-connectors-community              MySQL Connectors Community
mysql-tools-8.4-lts-community           MySQL Tools 8.4 LTS Community
```

Install MySQL

- MySQL 설치
- sudo yum install -y mysql-server



```
[ec2-user@ip-172-31-6-212 ~]$ sudo yum install -y mysql-server
MySQL 8.4 LTS Community Server                                   1.4 MB/s
MySQL Connectors Community                                       304 kB/s
MySQL Tools 8.4 LTS Community                                    760 kB/s
Dependencies resolved.
==========================================================================================
 Package                    Architecture    Version              Repository
==========================================================================================
Installing:
 mysql-community-server     x86_64          8.4.2-1.el9          mysql-8.4-lts-community
Installing dependencies:
 libaio                     x86_64          0.3.111-13.el9       baseos
 libtirpc                   x86_64          1.3.3-9.el9          baseos
```

- MySQL 설치 확인
- mysqld -V

```
[ec2-user@ip-172-31-6-212 ~]$ mysqld -V
/usr/sbin/mysqld  Ver 8.4.2 for Linux on x86_64 (MySQL Community Server - GPL)
```

Install MySQL

- MySQL 서버 실행 및 상태 확인
- sudo systemctl enable mysqld && sudo systemctl start mysqld && sudo systemctl status mysqld

```
[ec2-user@ip-172-31-6-212 ~]$ sudo systemctl status mysqld
● mysqld.service - MySQL Server
     Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; preset: disabled)
     Active: active (running) since Thu 2024-09-12 02:40:31 UTC; 24s ago
       Docs: man:mysqld(8)
             http://dev.mysql.com/doc/refman/en/using-systemd.html
    Process: 68400 ExecStartPre=/usr/bin/mysqld_pre_systemd (code=exited, status=0/SUCCESS)
   Main PID: 68465 (mysqld)
     Status: "Server is operational"
      Tasks: 35 (limit: 22622)
     Memory: 442.0M
        CPU: 2.545s
     CGroup: /system.slice/mysqld.service
             └─68465 /usr/sbin/mysqld
```

MySQL 기본

- MySQL 8.0 버전은 서버 설치과정에서 임시 비밀번호 생성
- 아래 명령어로 임시 비밀번호 확인
- sudo grep 'temporary password' /var/log/mysqld.log

```
d' /var/log/mysqld.log
A temporary password is generated for root@localhost: HAcEFTGLm8>l
```

MySQL 기본

- MySQL 접속
- mysql -u root -p
- 임시 비밀번호가 없는 경우 그냥 mysql 입력하면 됨.

```
[root@localhost shjeon]# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.30

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

MySQL 기본

- Root 유저의 비밀번호 변경
- 비밀번호에는 대문자, 소문자, 숫자, 기호 모두 조합
- ALTER USER 'root'@'localhost' IDENTIFIED BY 'Tibero1!';

```
[root@centos8 vagrant]# mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 12
Server version: 8.0.26 Source distribution

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> alter user 'root'@'localhost' IDENTIFIED BY 'TABA1';
Query OK, 0 rows affected (0.01 sec)
```

MySQL 기본

- 사용자 정보 확인
- select user, host, authentication_string FROM mysql.user;

```
mysql> select user, host, authentication_string from mysql.user;
+------------------+-----------+-------------------------------------------------------------------------+
| user             | host      | authentication_string                                                   |
+------------------+-----------+-------------------------------------------------------------------------+
| mysql.infoschema | localhost | $A$005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNEVERBRBEUSED |
| mysql.session    | localhost | $A$005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNEVERBRBEUSED |
| mysql.sys        | localhost | $A$005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNEVERBRBEUSED |
| root             | localhost | *9058A56D277B8F43DADC2E73AB1BEE8CDE7F9F37                               |
+------------------+-----------+-------------------------------------------------------------------------+
4 rows in set (0.00 sec)
```

MySQL 기본

- MySQL 내 데이터베이스 조회하기
- show databases;
- 데이터베이스 사용하기
- use database명

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.01 sec)

mysql> use mysql
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
```

MySQL 기본

- Database 내 테이블 확인하기
- show tables;

```
mysql> show tables;
+---------------------------------------------------------+
| Tables_in_mysql                                         |
+---------------------------------------------------------+
| columns_priv
| component
| db
| default_roles
| engine_cost
| func
| general_log
| global_grants
| gtid_executed
| help_category
| help_keyword
| help_relation
| help_topic
```

MySQL 기본

- Database 만들기
- create database 데이터베이스명;
- show databases;

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| testdb             |
+--------------------+
5 rows in set (0.01 sec)
```

데이터 정의어(Data Definition Language)

- 데이터 정의어(이하 DDL)는 데이터 간에 관계를 정의하여 데이터베이스 구조를 설정하는 SQL 문장

구분	명령어	설명
데이터베이스	CREATE DATABASE	데이터베이스를 생성한다.
	ALTER DATABASE	데이터베이스를 변경한다.
테이블	CREATE TABLE	테이블을 생성한다.
	ALTER TABLE	테이블을 변경한다.
	DROP TABLE	테이블을 제거한다.
테이블 스페이스	CREATE TABLESPACE	테이블 스페이스를 생성한다.
	ALTER TABLESPACE	테이블 스페이스를 변경한다.
	DROP TABLESPACE	테이블 스페이스를 제거한다.

데이터 정의어(Data Definition Language)

인덱스	CREATE INDEX	인덱스를 생성한다.
	ALTER INDEX	인덱스를 변경한다.
	DROP INDEX	인덱스를 제거한다.
뷰	CREATE VIEW	뷰를 생성한다.
	ALTER VIEW	뷰를 변경한다.
	DROP VIEW	뷰를 제거한다.
동의어	CREATE SYNOMYM	동의어를 생성한다.
	DROP SYNOMYM	동의어를 제거한다.
사용자	CREATE USER	사용자를 생성한다.
	ALTER USER	사용자를 변경한다.
	DROP USER	사용자를 제거한다.

데이터 정의어(Data Definition Language)

함수	CREATE FUNCTION	함수를 생성한다.
	ALTER FUNCTION	함수를 변경한다.
	DROP FUNCTION	함수를 제거한다.
프러시저	CREATE PROCEDURE	프러시저를 생성한다.
	ALTER PROCEDURE	프러시저를 변경한다.
	DROP PROCEDURE	프러시저를 제거한다.
타입	CREATE TYPE	타입을 생성한다.
	ALTER TYPE	타입을 변경한다.
	DROP TYPE	타입을 제거한다.

데이터 정의어(Data Definition Language)

권한	GRANT	사용자에게 특권을 부여한다.
	REVOKE	사용자에게 특권을 회수한다.
역할	CREATE ROLE	역할을 생성한다.
	ALTER ROLE	역할을 변경한다.
	DROP ROLE	역할을 제거한다.
객체	RENAME	테이블, 뷰, 동의어, 시퀀스 등의 스키마 객체의 이름을 변경한다.

Create Tables to Store Data

테이블 인스턴스 –테이블의 구조와 칼럼의 특성을 요약

Symbols	Explanation
PK	기본 키 열
FK	외래 키 열
FK1, FK2	동일한 테이블에 있는 두 개의 외부 키
NN	NOT NULL 열
U	고유 열

Create Tables to Store Data

MySQL에서 제공하는 데이터 타입

구분	데이터 타입
문자형	CHAR, VARCHAR, TEXT, MEDIUMTEXT, LONGTEXT, JASON
숫자형	TINYINT, SMALLINT, MEDIUMINT, INT, BIGINT, FLOAT, DECIMAL, DOUBLE
날짜형	DATE, TIME, DATETIME, TIMESTAMP, YEAR
이진 데이터 타입	BINARY, VARBINARY, TINYBLOB, BLOB,

Create Tables to Store Data

데이터 무결성 제약 조건

Constraint	Description
NOT NULL	NULL값을 허용하지 않고, 반드시 데이터를 입력. 제약조건이 NULL이면 해당 컬럼은 NULL값을 허용.
UNIQUE	해당 칼럼이 중복되는 데이터가 존재할 수 없는 유일성을 보장하는 제약조건
PRIMARY KEY	NOT NULL, UNIQUE 제약 조건의 결합과 같다. 테이블 또는 뷰는 단 한 개의 PRIMARY KEY 제약조건을 가질 수 있다.
FOREIGN KEY	같은 테이블 또는 서로 다른 두 개 테이블의 키 컬럼 사이의 관계

Create Tables to Store Data

Table Instance Chart

Table Name: EMPLOYEE

Column Name	Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
Key Type				PK					FK	
Nulls / Unique	NN		NN	NN						NN
FK Ref Table									EMPLOYEE	
FK Ref Column									Ssn	
Data Type	VARCHAR	CHAR	VARCHAR	CHAR	DATE	VARCHAR	CHAR	DECIMAL	CHAR	INT
Maximum Length	15		15	9		30		10,2	9	
Sample Data	John	B	Smith	123456789	1965-01-09	731 Foundren, Houston, TX	M	30000	333445555	5
	Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5

Create Tables to Store Data

Syntax

```
CREATE TABLE [user.] table_name
        ({column_name datatype | table_constraint
          ,column_name datatype | table_constraint});
```

where	*table_name*	테이블 이름
	column_name	열 이름
	datatype	데이터 타입
	table_constraint	제약 조건

Create Tables to Store Data

```
CREATE TABLE EMPLOYEE
( Fname          VARCHAR(10)   NOT NULL,
  Minit          CHAR,
  Lname          VARCHAR(20)     NOT NULL,
  Ssn            CHAR(9)         NOT NULL,
  Bdate          DATE,
  Address        VARCHAR(30),
  Sex            CHAR(1),
  Salary         DECIMAL(5),
  Super_ssn      CHAR(9),
  Dno            INT             NOT NULL,
PRIMARY KEY   (Ssn),
FOREIGN KEY (Super_ssn) REFERENCES EMPLOYEE (ssn));
```

Create Tables to Store Data

```
INSERT INTO EMPLOYEE
VALUES      ('James','E','Borg',888665555,'1937-11-10','450 Stone, Houston TX','M',55000,null,1),
            ('Franklin','T','Wong',333445555,'1965-12-08','638 Voss, Houston TX','M',40000,888665555,5),
            ('John','B','Smith',123456789,'1965-01-09','731 Fondren, Houston TX','M',30000,333445555,5),
            ('Ramesh','K','Narayan',666884444,'1962-09-15','975 Fire Oak, Humble TX','M',38000,333445555,5),
            ('Jennifer','S','Wallace',987654321,'1941-06-20','291 Berry, Bellaire TX','F',43000,888665555,4),
            ('Joyce','A','English',453453453,'1972-07-31','5631 Rice, Houston TX','F',25000,333445555,5),
            ('Alicia','J','Zelaya',999887777,'1968-01-19','3321 Castle, Spring TX','F',25000,987654321,4),
            ('Ahmad','V','Jabbar',987987987,'1969-03-29','980 Dallas, Houston TX','M',25000,987654321,4);
```

Create Tables to Store Data

EXAMPLE

Table Name: DEPARTMENT

Column Name	Dname	Dnumber	Mgr_ssn	Mgr_start_date
Key Type		PK	FK	
Nulls/Unique	NN,U	NN	NN	
FK Ref Table			EMPLOYEE	
Data Type	VARCHAR	INT	CHAR	DATE
Maximum Length	15		9	
Sample data	Research	5	333445555	1988-05-22

```
CREATE TABLE DEPARTMENT
( Dname          VARCHAR(15)      NOT NULL,
  Dnumber        INT              NOT NULL,
  Mgr_ssn        CHAR(9)          NOT NULL,
  Mgr_start_date  DATE,
PRIMARY KEY (Dnumber),
UNIQUE      (Dname),
FOREIGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn) );
```

Create Tables to Store Data

```
INSERT INTO DEPARTMENT
VALUES      ('Research',5,333445555,'1988-05-22'),
            ('Administration',4,987654321,'1995-01-01'),
            ('Headquarters',1,888665555,'1981-06-19');
```

Create Tables to Store Data

EXAMPLE

Table Name: DEPT_LOCATIONS

Column Name	Dnumber	Dlocation
Key Type	PK	PK, FK
Nulls/Unique	NN,U	NN
FK Ref Table		DEPARTMENT
FK Ref Column		Dnumber
Data Type	INT	VARCHAR
Maximum Length		15
Sample data	1	Houston

```
CREATE TABLE DEPT_LOCATIONS
( Dnumber       INT             NOT NULL,
  Dlocation     VARCHAR(15)     NOT NULL,
PRIMARY KEY (Dnumber, Dlocation),
FOREIGN KEY (Dnumber) REFERENCES DEPARTMENT(Dnumber) );
```

Create Tables to Store Data

```
INSERT INTO DEPT_LOCATIONS
VALUES      (1,'Houston'),
            (4,'Stafford'),
            (5,'Bellaire'),
            (5,'Sugarland'),
            (5,'Houston');
```

Create Tables to Store Data

EXAMPLE

Table Name: PROJECT

Column Name	Pname	Pnumber	Plocation	Dnum
Key Type		PK		FK
Nulls/Unique	NN	NN		NN
FK Ref Table				DEPARTMENT
FK Ref Column				Dnumber
Data Type	VARCHAR	INT	VARCHAR	INT
Maximum Length	15		15	
Sample data	ProductX	1	Bellaire	5

```
CREATE TABLE PROJECT
( Pname          VARCHAR(15)      NOT NULL,
  Pnumber        INT              NOT NULL,
  Plocation      VARCHAR(15),
  Dnum           INT              NOT NULL,
PRIMARY KEY (Pnumber),
UNIQUE      (Pname),
FOREIGN KEY (Dnum) REFERENCES DEPARTMENT(Dnumber) );
```

Create Tables to Store Data

```
INSERT INTO PROJECT
VALUES      ('ProductX',1,'Bellaire',5),
            ('ProductY',2,'Sugarland',5),
            ('ProductZ',3,'Houston',5),
            ('Computerization',10,'Stafford',4),
            ('Reorganization',20,'Houston',1),
            ('Newbenefits',30,'Stafford',4);
```

Create Tables to Store Data

EXAMPLE

Table Name: WORKS_ON

Column Name	Essn	Pno	Hours
Key Type	PK,FK	PK,FK	
Nulls/Unique	NN	NN	NN
FK Ref Table			
FK Ref Column			
Data Type	CHAR	INT	DECIMAL
Maximum Length	9		3,1
Sample data	123456789	1	32.5

```
CREATE TABLE WORKS_ON
( Essn          CHAR(9)         NOT NULL,
  Pno           INT             NOT NULL,
  Hours         DECIMAL(3,1)    NOT NULL,
PRIMARY KEY (Essn, Pno),
FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn),
FOREIGN KEY (Pno) REFERENCES PROJECT(Pnumber) );
```

Create Tables to Store Data

```
INSERT INTO WORKS_ON
VALUES     (123456789,1,32.5),
           (123456789,2,7.5),
           (666884444,3,40.0),
           (453453453,1,20.0),
           (453453453,2,20.0),
           (333445555,2,10.0),
           (333445555,3,10.0),
           (333445555,10,10.0),
           (333445555,20,10.0),
           (999887777,30,30.0),
           (999887777,10,10.0),
           (987987987,10,35.0),
           (987987987,30,5.0),
           (987654321,30,20.0),
           (987654321,20,15.0),
           (888665555,20,16.0);
```

Create Tables to Store Data

EXAMPLE

Table Name: DEPENDENT

Column Name	Essn	Dependent_name	Sex	Bdate	Relationship
Key Type	PK,FK	PK			
Nulls/Unique	NN	NN			
FK Ref Table	EMPLOYEE				
FK Ref Column	Ssn				
Data Type	CHAR	VARCHAR	CHAR	DATE	VARCHAR
Maximum Length	9	15			8
Sample data	3334455555	Alice	F	1986-04-05	Daughter

```
CREATE TABLE DEPENDENT
( Essn           CHAR(9)         NOT NULL,
  Dependent_name  VARCHAR(15)      NOT NULL,
  Sex            CHAR,
  Bdate          DATE,
  Relationship    VARCHAR(8),
PRIMARY KEY (Essn, Dependent_name),
FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn) );
```

Create Tables to Store Data

```
INSERT INTO DEPENDENT
VALUES      (333445555,'Alice','F','1986-04-04','Daughter'),
            (333445555,'Theodore','M','1983-10-25','Son'),
            (333445555,'Joy','F','1958-05-03','Spouse'),
            (987654321,'Abner','M','1942-02-28','Spouse'),
            (123456789,'Michael','M','1988-01-04','Son'),
            (123456789,'Alice','F','1988-12-30','Daughter'),
            (123456789,'Elizabeth','F','1967-05-05','Spouse');
```

CONFIRM THE STRUCTURE OF A TABLE

테이블 구조 확인

Syntax

DESCRIBE *table_name*

Example

mysql> DESCRIBE EMPLOYEE;

```
mysql> describe EMPLOYEE;
+-----------+--------------+------+-----+---------+-------+
| Field     | Type         | Null | Key | Default | Extra |
+-----------+--------------+------+-----+---------+-------+
| Fname     | varchar(10)  | NO   |     | NULL    |       |
| Minit     | char(1)      | YES  |     | NULL    |       |
| Lname     | varchar(20)  | NO   |     | NULL    |       |
| Ssn       | char(9)      | NO   | PRI | NULL    |       |
| Bdate     | date         | YES  |     | NULL    |       |
| Address   | varchar(30)  | YES  |     | NULL    |       |
| Sex       | char(1)      | YES  |     | NULL    |       |
| Salary    | decimal(5,0) | YES  |     | NULL    |       |
| Super_ssn | char(9)      | YES  | MUL | NULL    |       |
| Dno       | int          | NO   |     | NULL    |       |
+-----------+--------------+------+-----+---------+-------+
10 rows in set (0.00 sec)
```

ADD A COLUMN TO A TABLE

테이블에 칼럼 추가하기

Syntax

```
ALTER TABLE table_name
ADD ( column_name datatype
[, column_name datatype]…)
```

Example

```
ALTER TABLE DEPT_LOCATIONS
ADD COLUMN (comments VARCHAR (255));

DESCRIBE DEPT_LOCATIONS;
```

MODIFY A COLUMN

테이블에 존재하는 칼럼 속성 변경
데이터 타입, 기본값, 제약조건 변경

Syntax

```
ALTER TABLE table_name
MODIFY ( column_name datatype
[, column_name datatype]…)
```

Example

EMPLOYEE 테이블의 Fname 칼럼의 길이 20으로 변경 및 Bdate 값은 NULL이 될 수 없음

```
ALTER TABLE EMPLOYEE
MODIFY Fname varchar(20);

SQL> ALTER TABLE EMPLOYEE
MODIFY Bdate date NOT NULL;
```

ADD AND REMOVE DATA CONSTRAINTS

Example

s_emp 테이블의 제약조건 조회

```
select * from information_schema.table_constraints where table_name='EMPLOYEE';
```

```
mysql> select * from information_schema.table_constraints where table_name='EMPLOYEE';
+--------------------+-------------------+-----------------+--------------+------------+-----------------+-----------+
| CONSTRAINT_CATALOG | CONSTRAINT_SCHEMA | CONSTRAINT_NAME | TABLE_SCHEMA | TABLE_NAME | CONSTRAINT_TYPE | ENFORCED  |
+--------------------+-------------------+-----------------+--------------+------------+-----------------+-----------+
| def                | taba              | PRIMARY         | taba         | EMPLOYEE   | PRIMARY KEY     | YES       |
| def                | taba              | EMPLOYEE_ibfk_1 | taba         | EMPLOYEE   | FOREIGN KEY     | YES       |
+--------------------+-------------------+-----------------+--------------+------------+-----------------+-----------+
2 rows in set (0.00 sec)
```

DROP A TABLE

DROP TABLE 명령어를 사용해 데이터베이스에서 테이블 제거.
DROP TABLE 명령어는 한번 실행되면 취소할 수 없다.

Syntax

```
DROP TABLE table_name
```

Example

EX 예제 테이블 삭제

```
CREATE TABLE EX(ID INT);

DROP TABLE EX;
```

Simplify Data Access with Views

- 테이블 뷰를 만들어 논리적 하위 집합 또는 데이터 조합을 표시한다.
- 뷰는 실제 데이터가 포함되지 않는다.
- 뷰 이름은 테이블과 같은 네임스페이스를 사용하므로 스키마 내 다른 이름과 중복되면 안된다.

- 장점
  - 접근 제어로 보안 제공
  - 데이터 관리가 편리

- 단점
  - 삽입, 삭제, 갱신, 연산에 제약이 있다.

CREATE VIEWS

Syntax

```
CREATE VIEW view_name [ (alias, [alias]…)
AS query
WHERE
WITH CHECK OPTION
```

where	*view_name*	뷰 이름
	alias	별칭
	query	SELECT 문
	WITH CHECK OPTION	해당 옵션을 사용하면 INSERT/UPDATE 가능
	WITH READ ONLY	읽기 전용 뷰
	constraint	CHECK OPTION에 할당된 제약조건

CREATE VIEWS

Example

부서 번호가 5인 직원의 SSN, 이름이 포함된 뷰를 생성

```
CREATE VIEW EMP5
    AS SELECT Ssn, Fname, Lname
    FROM EMPLOYEE
    WHERE Dno = 5;


select * from EMP5;

+-----------+----------+---------+
| Ssn       | Fname    | Lname   |
+-----------+----------+---------+
| 123456789 | John     | Smith   |
| 333445555 | Franklin | Wong    |
| 453453453 | Joyce    | English |
| 666884444 | Ramesh   | Narayan |
+-----------+----------+---------+
4 rows in set (0.01 sec)
```

CREATE VIEWS

Example

부서 번호가 4인 뷰를 생성. Ssn은 ID , Fname은 Employee로 표현.

```
CREATE VIEW EMP4 (ID, Employee)
    AS SELECT Ssn, Fname
    FROM EMPLOYEE
    WHERE Dno = 4;

mysql> select * from EMP4;

+-----------+----------+
| ID        | Employee |
+-----------+----------+
| 987654321 | Jennifer |
| 987987987 | Ahmad    |
| 999887777 | Alicia   |
+-----------+----------+
3 rows in set (0.00 sec)

5 rows selected.
```

CREATE VIEWS

Example

월급이 30,000 이상인 뷰를 생성. 모든 정보를 포함

```
CREATE VIEW SAL30000
    AS SELECT *
    FROM EMPLOYEE
    WHERE Salary >= 30000;

SQL> SELECT *
   FROM SAL30000;

+----------+-------+---------+-----------+------------+------------------------+------+--------+-----------+-----+
| Fname    | Minit | Lname   | Ssn       | Bdate      | Address                | Sex  | Salary | Super_ssn | Dno |
+----------+-------+---------+-----------+------------+------------------------+------+--------+-----------+-----+
| John     | B     | Smith   | 123456789 | 1965-01-09 | 731 Fondren, Houston TX | M    |  30000 | 333445555 |   5 |
| Franklin | T     | Wong    | 333445555 | 1965-12-08 | 638 Voss, Houston TX    | M    |  40000 | 888665555 |   5 |
| Ramesh   | K     | Narayan | 666884444 | 1962-09-15 | 975 Fire Oak, Humble TX | M    |  38000 | 333445555 |   5 |
| James    | E     | Borg    | 888665555 | 1937-11-10 | 450 Stone, Houston TX   | M    |  55000 | NULL      |   1 |
| Jennifer | S     | Wallace | 987654321 | 1941-06-20 | 291 Berry, Bellaire TX  | F    |  43000 | 888665555 |   4 |
+----------+-------+---------+-----------+------------+------------------------+------+--------+-----------+-----+
```

DROP A VIEW

Syntax

```
DROP VIEW view_name
```

Example

EMP5 뷰 삭제

```
DROP VIEW EMP5;
```

Control User Access: Overview

- 데이터베이스 관리자는 사용자에게 **SQL** 보안 명령을 사용해 테이블에 대한 액세스 권한을 제공

- Control User Access
  - 데이터베이스에 대한 권한 제공
  - 테이블같은 사용자 개체에 대한 액세스를 제공하고 제거
  - 데이터 사전에서 주어진 권한 및 받은 권한 확인

SYSTEM PRIVILEGES: OVERVIEW

- 데이터베이스 관리자는 사용자에게 시스템 권한을 부여하여 사용자는 특정 작업을 수행할 수 있다.
- 시스템 권한은 명령을 실행할 수 있는 권한이다.

시스템 권한 유형

System Privilege	Description
In One's Own Schema	자신의 스키마에 테이블 및 시퀀스를 생성할 수 있는 권한
On all Objects of a Specified Type	모든 스키마에서 테이블 생성 및 테이블 또는 뷰를 업데이트 할 수 있는 권한
On the System or a User	사용자를 생성할 수 있는 권한

SYSTEM PRIVILEGES: OVERVIEW

- 60개가 넘는 고유한 시스템 권한이 있다.
- 각 시스템 권한을 통해 사용자는 특정 작업을 수행할 수 있다.

시스템 권한 유형

Class	System Privilege	Operations Permitted
SESSION	CREATE SESSION	데이터베이스 연결 허용
TABLE	CREATE TABLE	테이블 및 인덱스 생성
		CONNECT, DML, DROP, ALTER, TRUNCATE 가능
TABLE	SELECT ANY TABLE	모든 스키마에 모든 테이블, 뷰 쿼리 사용 가능

GRANT SYSTEM PRIVILEGES

- 데이터베이스 관리자는 **GRANT SQL** 명령을 사용하여 사용자 및 역할 권한을 부여할 수 있다.

Syntax

GRANT *system_priv* ON *Database.table* TO [*user, role*] @'localhost'

where	*system_priv*	부여되는 시스템 권한
	Database	데이터베이스 명
	table	테이블 명
	TO	권한을 부여 받는 대상
	user	일반 사용자
	role	권한 받을 역할

GRANT SYSTEM PRIVILEGES

Example

MySQL에 등록된 사용자 확인

```
use mysql;

select user, host from user;

+------------------+-----------+
| user             | host      |
+------------------+-----------+
| mysql.infoschema | localhost |
| mysql.session    | localhost |
| mysql.sys        | localhost |
| root             | localhost |
+------------------+-----------+
```

GRANT SYSTEM PRIVILEGES

Example

실습을 위해 사용자 생성

```
CREATE USER 'USER명'@'localhost' IDENTIFIED BY 'Password'; (password 대/소문자, 숫자, 기호 포함)

CREATE USER 'scott'@'localhost' IDENTIFIED BY 'Tibero1!';
```

GRANT SQL 명령을 사용해 scott 사용자가 taba 데이터베이스에서 CREATE를 할 수 있는 권한 부여

```
GRANT CREATE ON taba.* to 'scott'@'localhost';
```

scott 사용자가 taba 데이터베이스에서 SELECT, INSERT를 할 수 있는 권한 부여

```
GRANT SELECT, INSERT ON taba.* to 'scott'@'localhost';
```

GRANT OBJECT PRIVILEGES

Example

scott에게 EMPLOYEE 테이블을 조회 할 수 있는 권한 부여

```
GRANT SELECT ON taba.EMPLOYEE TO 'scott'@'localhost';
```

scott에게 EMPLOYEE 테이블의 급여를 수정할 수 있는 권한 부여

```
GRANT UPDATE(Salary)
ON taba.EMPLOYEE
TO 'scott'@'localhost';
```

GRANT OBJECT PRIVILEGES

Example

scott에게 DEPARTMENT 테이블을 조회 할 수 있는 권한과 다른 사람에게 동일한 권한을 부여할 수 있는 권한 부여

```
GRANT SELECT
ON taba.DEPARTMENT
TO 'scott'@'localhost'
WITH GRANT OPTION;
```

현재 사용자가 scott에게 부여된 객체 조회

```
+-----------------------------------------------------------------------------+
| Grants for scott@localhost                                                  |
+-----------------------------------------------------------------------------+
| GRANT USAGE ON *.* TO `scott`@`localhost`                                   |
| GRANT SELECT, INSERT, UPDATE, CREATE ON `taba`.* TO `scott`@`localhost`     |
| GRANT SELECT ON `taba`.`DEPARTMENT` TO `scott`@`localhost` WITH GRANT OPTION |
| GRANT SELECT, UPDATE (`Salary`) ON `taba`.`EMPLOYEE` TO `scott`@`localhost` |
+-----------------------------------------------------------------------------+
```

REMOVE OBJECT PRIVILEGES

- 데이터베이스 관리자는 **REVOKE SQL** 명령을 사용하여 사용자에게 부여된 권한을 제거

Syntax

REVOKE *system_priv* ON *Database.table* FROM [*user, role*] @'localhost'

where	*system_priv*	부여되는 시스템 권한
	Database	데이터베이스 명
	table	테이블 명
	FROM	권한을 제거할 대상
	user	일반 사용자
	role	권한 받을 역할

REMOVE OBJECT PRIVILEGES

scott에게 DEPARTMENT 테이블을 조회할 수 있는 권한 제거

Example

```
REVOKE SELECT
   ON taba.DEPARTMENT
   FROM 'scott'@'localhost';
```

scott의 모든 권한을 제거

Example

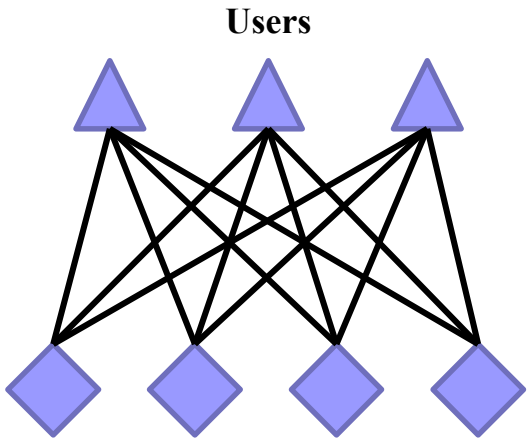
```
REVOKE ALL
ON *.*
FROM 'scott'@'localhost';

SHOW GRANTS FOR 'scott'@'localhost';
```

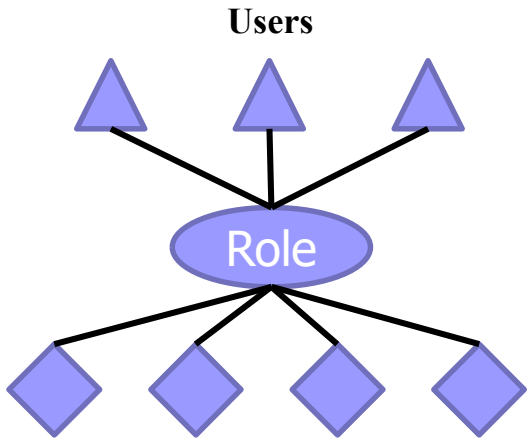
PRIVILEGES GROUPED BY ROLE

- 역할 사용을 통해 권한 관리를 단순화 시킴
- 시스템 권한과 객체 권한으로 구성 가능
- 역할은 사용자가 소유하지 않고, 스키마에도 존재하지 않음.

Grant Privileges Without Roles



Grant Privileges With Roles



CREATE A ROLE

Syntax

```
CREATE ROLE role_name;
```

Example

manager 역할 생성

```
CREATE ROLE manager;
```

manager역할에 EMPLOYEE 테이블의 Salary칼럼 수정, 데이터값 삽입이 가능하도록 권한 부여

```
GRANT UPDATE(Salary), INSERT
ON taba.EMPLOYEE
TO manager;
```

scott에게 manager 권한 부여

```
GRANT manager
TO 'scott'@'localhost';
```