




데이터베이스(sql/nosql)

DB 수업내용



- DB소개
- RDBMS 이해
- SQL실습환경 셋팅
- SQLite로 SQL 기본 익히기
- **mysql 설치, 설정, 사용자 생성**
- **SQL(with mysql)**
 - CML(CRUD), DDL
 - Join, SQL function
- Nosql(mongoDB)



학습목표 – mysql DB

Ubuntu Linux에 Mysql 서버를 설치한다.

Mysql 서버 실행, 종료를 한다.

Mysql DB 개발자 계정을 생성하고, 접속 권한을 부여한다.

Dbeaver로 Mysql을 연결하여 SQL CRUD로 데이터를 조작한다.

SQL Join, SQL의 다양한 함수를 안다.

MySQL 설치 및 실습

Mysql 셋팅 과정

- mysql 서버 설치 / 실행 / 종료
- mysql 프로세스 실행 확인
- mysql 보안 레벨 셋팅(mysql root 계정 passwd, 보안 등)
- 추가 패키지 설치
 - mysql python 패키지 설치
- root 계정으로 DB 접속
 - mysql DB접속 개발자 계정 만들기
 - 개발자 계정에 권한 부여
- 개발자 계정으로 DB 접속 / DB 확인

MySQL 설치 및 실습

2-1mysql_basic_install_setting.pdf 참조

Mysql와 Deaver와 연결

- server host / port
- database
- id / passwd



Connect to a database

MySQL connection settings

MySQL logo

Main Driver properties SSH Proxy SSL

Server

Server Host: localhost Port: 3306

Database:

Authentication (Database Native)

Username: cozlab

Password: ☒ Save password locally

Advanced

Server Time Zone: Auto-detect

Local Client: <not present>

i You can use variables in connection parameters. [Connection details \(name, type, ...\)](#)

Driver name: MySQL [Edit Driver Settings](#)

Test Connection ... < 이전(B) 다음(N) > 취소 완료(F)

Deaver – universal DB Client

The screenshot displays the DBeaver 21.2.1 - sqlDB interface. The top menu bar includes options like 파일(F), 편집(E), 탐색(N), 검색(A), SQL 편집기, 데이터베이스(D), 윈도우(W), and 도움말(H). The toolbar contains various icons for file operations, SQL editing, and database management. The left sidebar shows a tree view with 'Databases' expanded, listing 'localhost - localhost:3306' and its sub-items: 'Databases', 'Users', 'Administer', and 'System Info'. The main panel is divided into two tabs: 'Properties' and '엔티티 관계도'. The 'Properties' tab is active, showing fields for 'Schema Name' (sqlDB), 'SQL Path', 'Default Charset' (utf8), 'Database size' (48K), and 'Default Collation' (utf8_bin). Below the properties, a table lists database objects:

테이블명	Engine	Auto Increment	Data Length	Description
buyTbl	InnoDB	23	16K	
userTbl	InnoDB	0	16K	

The bottom status bar shows '2 items' and a toolbar with icons for search, filter, settings, and other actions. The bottom-most bar displays 'KST ko_KR'.

SQL DDL 학습 내용



- 데이터 타입
- CREATE
- ALTER
- DROP

MySQL – SQL

DML; Data Manipulation Language

MySQL 데이터 유형과 타입 종류

- 데이터 유형과 타입 종류

구분	데이터 타입 종류
문자형	CHAR, VARCHAR, BLOB
숫자형	INT, BIGINT, NUMERIC, FLOAT, DOUBLE
날짜형	DATE, TIMESTAMP
논리형	BOOLEAN(TRUE/FALSE)
null	null값(데이터 없음)
BLOB	바이너리 데이터(image, video, audio, binary 파일)

학습내용



- INSERT
- SELECT
- UPDATE
- DELETE
- Group BY
- Where
- Join

실습1



- sql 스크립트 파일
sql_mysql_basic_DML01_p.sql 활용

엑셀파일 DB로 import 하기

[실습]SQL 더 나아가기

- 거래액 데이터 분석
- 매출이 높은 주요 카테고리만 확인하기
- 관련 csv 파일 db로 import 하기

데이터 출처 :

<https://kosis.kr/index/index.do>

온라인 쇼핑 거래액

테이블명:

category

Engine:

InnoDB

Auto Increment:

0

Charset:

utf8

Collation:

utf8_general_ci

Description:

Columns

Constraints

Foreign Keys

References

Triggers

Indexes

Partitions

Statistics

DDL

Virtual

컬럼명	#	Data Type	Not Null	Auto Increment	Key	Extra	Expression	Comment
id	1	int(11)	[]	[]				
cate1	2	varchar(5)	[]	[]				
cate2	3	varchar(4)	[]	[]				
cate3	4	varchar(6)	[]	[]				

DB에 데이터 import

[illegible]

© 2006 The Authors
Journal compilation © 2006 Blackwell Publishing Ltd

DB에 데이터 import

테이블명:

Engine:

Auto Increment:

Charset:

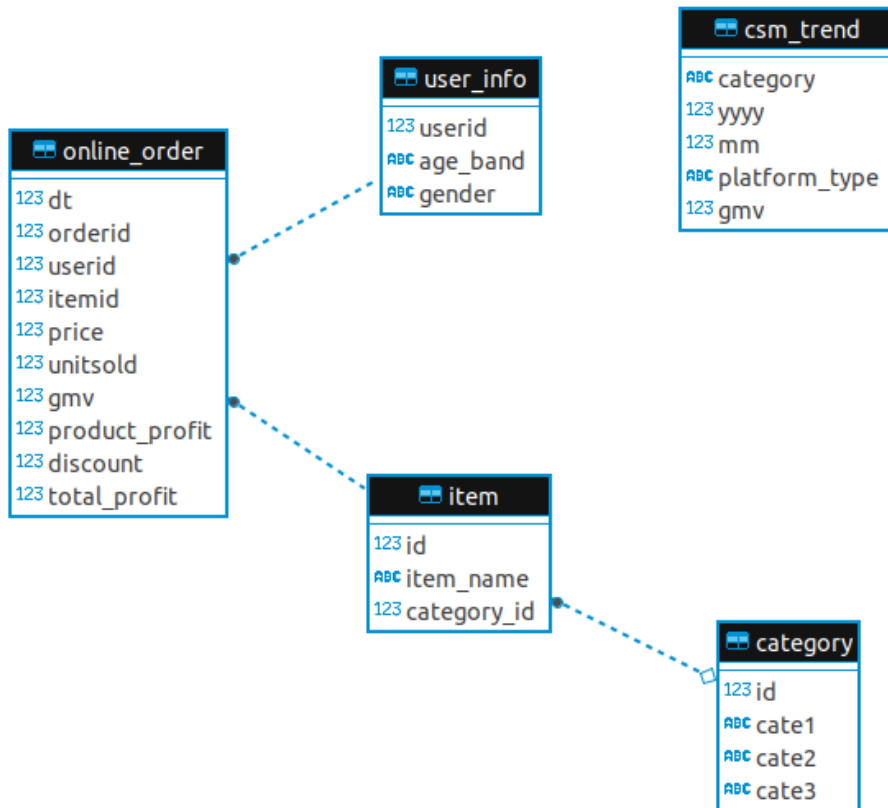
Collation:

Description:

Columns	컬럼명	#	Data Type	Not Null	Auto Increment	Key	Extra	Expression	Comment
Constraints	123 dt	1	int(11)	[]	[]				
Foreign Keys	123 orderid	2	int(11)	[]	[]				
References	123 userid	3	int(11)	[]	[]				
Triggers	123 itemid	4	int(11)	[]	[]				
Indexes	123 price	5	int(11)	[]	[]				
Partitions	123 unitsold	6	int(11)	[]	[]				
Statistics	123 gmv	7	int(11)	[]	[]				
DDL	123 product_profit	8	int(11)	[]	[]				
Virtual	123 discount	9	int(11)	[]	[]				
	123 total_profit	10	int(11)	[]	[]				

[illegible]

ERD 그리기



Where절과 Having절 정리

Where절과 Having절의 차이

```
select category, sum(gmv) as tot_gmv  
from gmv_trend  
group by 1  
having sum(gmv_) >= 40000000
```

```
select category, sum(gmv) as tot_gmv  
from gmv_trend  
where yyyy = 2020  
group by 1  
having sum(gmv_) >= 10000000
```

Where절과 Having절의 차이

- Where절과 Having절의 차이

where

집계 전 데이터를 필터링

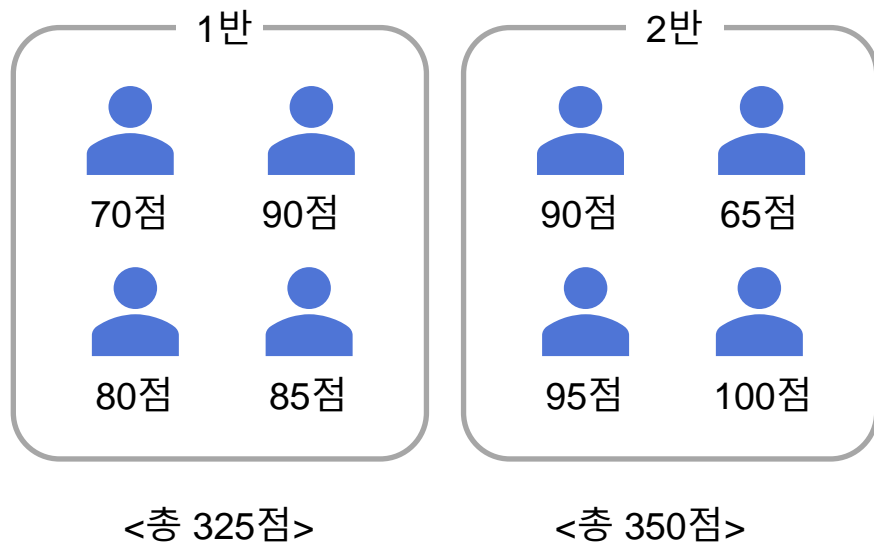
having

집계 후 데이터를 필터링

where 절에는 집계함수가 올 수 없고,
having 절에는 집계함수 만 올 수 있음

Where절과 Having절의 차이

- where 조건을 주었을 때 vs having 조건을 주었을 때



where 조건

```
select *  
from stdnt_score  
where score >=70
```

Having 조건

```
select class_num, sum(score) as tot_score  
from stdnt_score  
group by class_num  
having sum(score) >= 350
```

Where절과 Having절의 차이

where 조건

```
select *  
from stdnt_score  
where score >=70
```

1반



70점



90점



80점



85점

<총 325점>

2반



90점



95점



100점

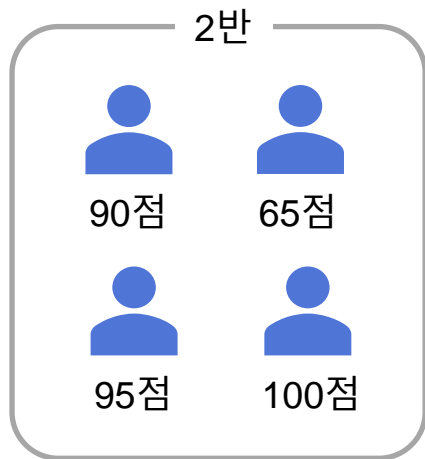
<총 350점>

집계 전 단계에서
필터링

Where절과 Having절의 차이

Having 조건

```
select class_num, sum(score) as tot_score  
from stdnt_score  
group by class_num  
having sum(score) >= 350
```

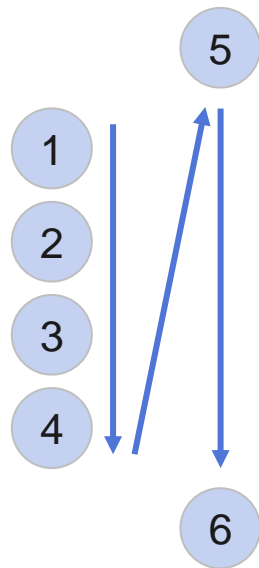


<총 350점>

집계 후 단계에서
필터링

SQL 문법 정리

select [컬럼명], ...
from [테이블명]
where [조건절]
group by [컬럼명 or 컬럼순서]
having [집계함수 조건절]
order by [컬럼명 or 컬럼순서]
limit[N]



SQL 문법 정리 - select절 주의사항

- select 절에서 가장 많이 하는 실수
- 컬럼명을 새로 명명할 경우,
- 띄어쓰기, 숫자로 시작하는 단어, 특수문자 사용 못함.
- 큰 따옴표(")로 감싸준 경우, 모두 가능함.(권장하지 않음)

잘못된
예시

```
select category as cate gory, gmv as 2021_gmv, mm as #mm  
from csm_trend  
where yyyy = 2021
```

가능한
예시

```
select category as "cate gory", gmv as "2021_gmv", mm as "#mm"  
from csm_trend  
where yyyy = 2021
```

SQL 문법 정리 - where절 조건 정리

긍정	부정	의미
=	!= 또는 <>	일치 / 불일치 조건
>, >=, <, <=		비교 조건(보통 숫자와 많이 사용)
between ~ and ~	not between ~ and b	범위 조건(보통 숫자와 많이 사용)
in (~, ~, ~)	not in (~, ~)	다수의 값 일치 / 불일치 조건
like '%~%'	not like '%~%'	특정 문자열 포함 / 불포함 조건
ilike '%~%'	not ilike '%~%'	위와 동일, 영어인 경우 알파벳 대소문자 구분 안함
is null	is not null	null값 포함 / 제외 조건

SQL 문법 정리 - where절 주의 사항

- where절에 and와 or를 함께 쓸 때는, 반드시 의미단위로 ()로 묶기

조건

매출액이 100만원 이상이거나
판매 수량이 50개 이상이면서,
식음료 카테고리 여야 함.

성격이 좋거나 착하면서,
유머코드가 잘 통하는 사람.

Good Case

(매출액 >= 100만원 or 판매수량 >= 50)
and
카테고리 = '식료품'

(외모 준수 or 성격 좋음)
and
유머코드 잘 맞음

SQL 문법 정리 – group by절 주의 사항

- select 절에 등장한 컬럼이 group by에 모두 명시되지 않은 경우(집계함수 제외)

```
SELECT category cate, yyyy year, sum(gmv) total_gmv
FROM csm_trend
group by category
```

csn_trend 1

#SELECT category as cate, yyyy year, sum(gmv) total_gmv

SQL Error [1055] [42000]: Expression #2 of SELECT list is not in GROUP BY clause and contains nonaggregated column 'mysqldb.csm_trend.yyyy' which is not functionally dependent on columns in GROUP BY clause; this is incompatible with sql_mode=only_full_group_by

세부사항(D) >>

!!! 주의

집계함수 빼고는 모두 group by에 명시해야 함.

집계함수 앞의 컬럼을 추가하거나 뺄 때 group by 절도 같이 수정해 줘야함.

Table Join 살펴보기

Join이 필요한 이유

- 필요한 정보(상품명, 유저 정보, 카테고리 정보 등) 각기 다른 테이블에 분산 저장 되어 있을 때, 이를 하나의 테이블로 추출하기 위해
- join의 위치 및 문법

```
select 컬럼명  
from 테이블명 (as) t1  
join 테이블명2 (as) t2 on t1.컬럼명 = t2.컬럼명
```

[주의] 두 개이상의 테이블에 동일한 컬럼명이 있을 때, select절이나 where절에 그 컬럼명을 사용할 경, **t1.컬럼명** 과 같이 어떤 테이블로부터 가져올 것인지 구체적으로 명시할 것

데이터 설명

- 주문테이블 컬럼 상세 설명(online_order)

컬럼명	의미	수식
price	판매가격	
unitsold	판매수량	
gvm	거래액	$\text{unitsold} * \text{price}$
product_profit	상품이익	
discount	할인액	
total_profit	총 이익	$\text{product_profit} - \text{discount}$

Join문 문법

기본 형식

```
select 컬럼명  
from 테이블명 (as) a  
join 테이블명2 (ab) b on a.컬럼명 = b.컬럼명
```

예

```
select itemid, item_name, sum(gmv) as tot_gmv  
from online_order as oo  
join item as i on oo.itemid = i.id  
group by 1, 2  
order by 3 desc
```

Join문 문법

표준 sql(ansi sql – rdbms 표준)

기본 형식

```
select 컬럼명  
from 테이블명1 (as) a  
join 테이블명2 (as) b on a.컬럼명 = b.컬럼명  
where 조건
```

예

```
select item_name, sum(gmv) as tot_gmv  
from online_order as oo  
join item as i on oo.itemid = i.id  
join user_info ui on oo.userid = ui.userid  
where gender = 'M'  
group by 1  
order by 2 desc
```

Join문 문법

표준 sql
(ansi sql,
rdbms 표준)

```
select 컬럼명  
from 테이블명1 (as) a  
join 테이블명2 (as) b on a.컬럼명 = b.컬럼명  
where 조건
```

오라클 방식
join 문법

```
select a.컬럼명  
from 테이블명1 (as) a, 테이블명2 (as) b  
where a.컬럼명 = b.컬럼명  
and 조건1  
and 조건2
```

Join문 살펴보기

주문 테이블에는 user id가 null인 경우가 있다.
 >> Question. user 테이블을 조인하면 어떻게 될까?

123 dt	123 orderid	123 userid	123 itemid	123 price
20,210,601	37,453,043	514,972	1	20,1
20,210,601	48,290,697	191,515	1	20,1
20,210,601	63,666,257	787,667	1	20,1
20,210,601	96,701,451	500,786	1	20,1
20,210,601	11,275,835	143,232	1	20,1
20,210,601	11,417,271	520,163	1	20,1
20,210,601	11,941,351	638,354	1	20,1
20,210,601	11,996,211	442,878	1	20,1
20,210,601	12,751,862	621,174	1	20,1
20,210,601	13,231,881	307,243	1	20,1
20,210,601	13,249,255	[NULL]	1	20,1
20,210,601	13,514,650	[NULL]	1	20,1
20,210,601	13,912,399	[NULL]	1	20,1
20,210,601	13,986,809	[NULL]	1	20,1
20,210,601	14,277,352	540,098	1	20,1
20,210,601	14,564,054	387,719	1	20,1
20,210,601	14,690,277	592,623	1	20,1

123 userid	123 age_band	123 gender
507,734	20~24	F
280,456	20~24	F
401,841	20~24	F
708,361	20~24	F
354,070	20~24	F
832,796	20~24	F
514,972	20~24	F
191,515	20~24	F
787,667	20~24	F
500,786	20~24	F
143,232	20~24	F
520,163	20~24	F

(1) 해당 행이 삭제된다.

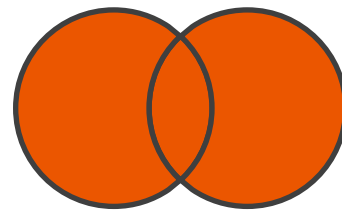
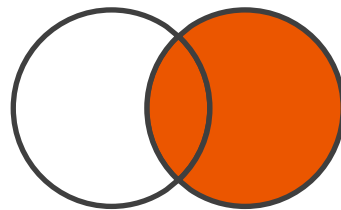
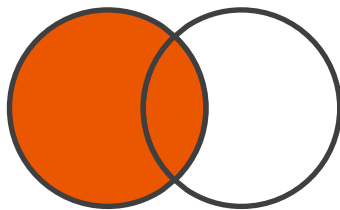
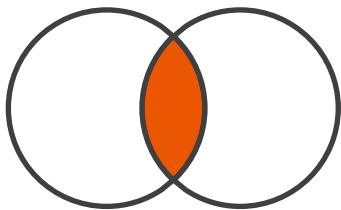
123 dt	123 orderid	123 userid	123 age_band	123 gender
20,210,601	83,515,822	708,361	20~24	F
20,210,601	83,575,844	354,070	20~24	F
20,210,601	17,604,107	832,796	20~24	F
20,210,601	37,453,043	514,972	20~24	F

(2) 해당 행이 그대로 남는다.
 단, 아무 정보도 가져오지 못한다.

123 dt	123 orderid	123 userid	123 age_band	123 gender
20,210,601	11,941,351	638,354	20~24	F
20,210,601	11,996,211	442,878	20~24	F
20,210,601	12,751,862	621,174	20~24	F
20,210,601	13,231,881	307,243	20~24	F
20,210,601	13,249,255	[NULL]	[NULL]	[NULL]
20,210,601	13,514,650	[NULL]	[NULL]	[NULL]
20,210,601	13,912,399	[NULL]	[NULL]	[NULL]
20,210,601	13,986,809	[NULL]	[NULL]	[NULL]
20,210,601	14,277,352	540,098	20~24	F
20,210,601	14,564,054	387,719	20~24	F
20,210,601	14,690,277	592,623	20~24	F

Join의 유형과 상황별 예시

	Inner Join	Left Join	Right Join	Full Join
조인 결과	두개의 테이블에 모두 존재하는 행만 남음	왼쪽 테이블을 기준으로 오른쪽 테이블을 붙임	Left join과 반대	모든 값이 합쳐짐
필요 상황	두개의 테이블에 조인 키가 빠짐없이 있을 때	한 개 이상 테이블의 조인키에 null 값이 있을 때	없음	거의 없음



Join 유형과 상황 별 예시

- inner join select ui.gender, ui.age_band, sum(gmv) as tot_gmv
 from online_order oo
 (inner) join user_info ui on oo.userid = ui.userid
 group by 1, 2
 order by 1, 2
- left join select ui.gender, ui.age_band, sum(gmv) as tot_gmv
 from online_order oo
 left join user_info ui on oo.userid = ui.userid
 group by 1, 2
 order by 1, 2

Join 주의사항 – 많이 하는 실수

- 행 중복 case

item_id	item_name	category_id
123345	득템 원피스	443
124543	깜찍 블라우스	765
125345	필수템 셔츠	986



category_id	category
443	원피스류
765	블라우스류
986	셔츠류
986	정장류



item_id	item_name	category_id	category
123345	득템 원피스	443	원피스류
124543	깜찍 블라우스	765	블라우스류
125345	필수템 셔츠	986	셔츠류
125345	필수템 셔츠	986	정장류

조인 키에 특정 테이블에 의도치 않게 중복값이 있으면, 조인 결과 중복이 발생한다. (이를 알려주지 않음!)

SQL 함수 살펴보기

Sql 함수1 - 추출

- cast()
- left()
- substring()
- right()
- concat() , ||
- coalesce()
- case when ~ then ~ then ~ else ~ end as 컬럼명
- replace()

Sql 함수1 - 날짜

날짜 관련 함수

- 현재 시간: `now()`, `current_date`, `current_timestamp`
- `to_char(now(), 'yyyymmdd')`
- `to_char(now(), 'yyyy-mm-dd')`
- `now() + interval '1 day'`
- `now() - interval '1 month'`
- `dateadd('month', -2, now())` # 현재기준 2개월 전
- `date_part('month', now())` # 'year', 'month', 'day'

Sql 함수1

예) 최근 1년 동안의 매출액 확인

```
select *  
from gmv_trend gt |  
where cast(yyyy as varchar) || cast(mm as varchar)  
>= cast(date_part('year', now() - interval ]1 year]) as varchar) ||  
cast(date_part('month', now() - interval '1 year') as varchar)  
order by 2, 3
```

SQL 응용문법 정리

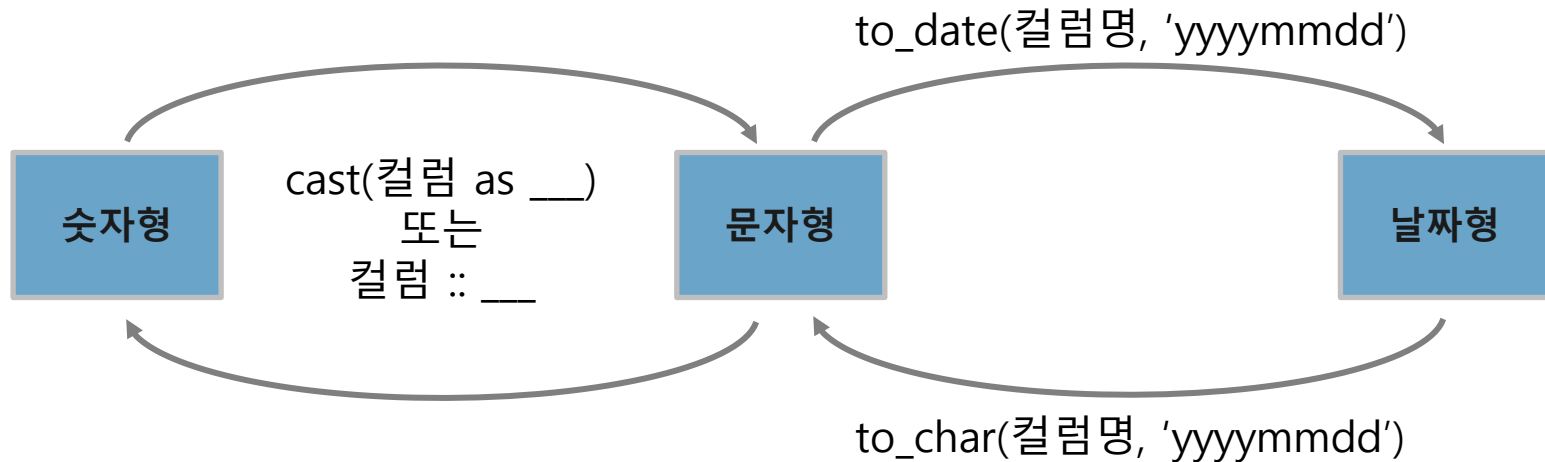
Sql 데이터 유형

- 데이터의 유형과 종류

구분	데이터 타입 종류
문자형	CHAR, VARCHAR, LOGN, CLOB
숫자형	INT, BIGINT, NUMERIC, FLOAT, DOUBLE
날짜형	DATE, TIMESTAMP
논리형	BOOLEAN(TRUE/FALSE)

데이터 가공 문법 정리

- 데이터 유형 변환



데이터 가공 문법 정리

- 문자형 데이터 가공하기

구분	명령 종류	
특정 문자열만 잘라내기	left, right, substring	
문자열 합치기	컬럼A 컬럼B	concat(컬럼A, 컬럼B)
조건별로 그룹핑하기	case when ~ else ~ end	
Null값 치환하기	coalesce(컬럼명, ~~~)	nvl(컬럼명, ~~~)
특정 문자열 자르기	trim, ltrim, rtrim	
특정 문자열 치환하기	replace("원래문자", "바꿀문자")	
대문자로 변경	upper	
소문자로 변경	lower	
문자열의 길이 반환	length, len	

데이터 가공 문법 정리

- 숫자형 데이터 가공하기

구분	명령 종류	
반올림	round	
올림	ceiling	
내림	trunc	
절대값 반환	abs	
Null 값 치환	coalesce(컬럼명, 0)	nul(컬럼명, 0)

데이터 가공 문법 정리

• 날짜 데이터 가공하기

구분	명령 종류	
날짜 더하기	날짜 - interval '1 day'	dateadd('day', 01, 날짜)
날짜간 빼기	datediff('day', 시작날짜, 종료날짜)	
특정 월/주/분기 등 추출	date_part('month', 날짜)	to_char(날짜, 'mm')
<ul style="list-style-type: none">- 현재 날짜의 월 1일- 현재날짜의 분기 1일- 주 시작일 구하기	<ul style="list-style-type: none">date_trunc('month', now())date_trunc('quarter', now())date_trunc('week', now())	

MySQL – SQL

DDL(Data Definition Language)

MySQL 데이터 유형과 타입 종류

- 데이터 유형과 타입 종류

구분	데이터 타입 종류
문자형	CHAR, VARCHAR, BLOB
숫자형	INT, BIGINT, NUMERIC, FLOAT, DOUBLE
날짜형	DATE, TIMESTAMP
논리형	BOOLEAN(TRUE/FALSE)
null	null값(데이터 없음)
BLOB	바이너리 데이터(image, video, audio, binary 파일)

MySQL – DDL 실습



- sql_mysql_basic_ddl_p.sql 파일 활용
- sql_mysql_basic_ddl_ps.sql 결과 파일 – 수업 후 배포

실습

크롤링한 데이터를 DB에 저장하기