함수

숫자를 다루는 함수

올림, 내림, 반올림 - `FLOOR()`, `CEIL()`, `ROUND()`

- `FLOOR(x)`
  - 내림 (해당 필드 내의 값보다 작은 최대의 정수 값을 반환)
- `CEIL(x)`, `CEILING(x)`
  - 올림 (해당 필드 내의 값 이상의 최소 정수 값을 반환)
- `ROUND(x, digit=0)`
  - 해당 필드 내의 값을 지정한 자릿수에서 반올림 (丸める)
    - 자릿수는 생략 가능하며 , 기본값은 1의 자릿수에서 반올림
  - 자릿수가 양수일 경우 소수부, 음수일 경우 정수부에서 반올림
    - `ROUND(123.45, 0)` = 123.0
    - `ROUND(123.45, 1)` = 123.5
    - `ROUND(123.45, -1)` = 120.0

```
SELECT
  num,
  FLOOR(num) AS floor,
  CEIL(num) AS ceil,
  ROUND(num) AS round_1,
  ROUND(num, 1) AS round_2,
  ROUND(num, -1) AS round_3
FROM (
  SELECT 123.456 AS num
);

-- RESULT;
-- | num     | floor | ceil  | round_1 | round_2 | round_3 |
-- | ------- | ----- | ----- | ------- | ------- | ------- |
-- | 123.456 | 123.0 | 124.0 | 123.0   | 123.5   | 120.0   |
--
```

절댓값, 나머지 연산 - `ABS()`, `MOD()`

- `ABS(x)`
  - 주어진 숫자의 절댓값을 구한다.
  - 예시) 행을 절댓값으로 정렬
- `MOD(x, y)`
  - x를 y로 나눈 나머지를 구한다.
  - 예시) 홀수/짝수만 탐색

난수 생성 - `RAND()`

- `rand()`
  - 0 이상 1 미만의 값을 생성한다
  - `ORDER BY RAND()`

문자열을 다루는 함수

- `CONCAT(x[, ...])`
  - 인수로 온 값을 연결함
- `SUBSTR(x, pos[, count])`
  - 문자열의 위치에서 주어진 숫자만큼만 추출한다.
  - 첫 번째 문자의 인덱스는 1이며, count가 2일경우 두 문자를 추출
  - pos만 주어졌을 경우 pos번째(포함) 부터 끝까지 추출
    - pos는 음수로 입력할 수 있으며, 이 경우 뒤에서 n번째 글자부터 끝까지 추출
    - count는 음수일 수 없음
- `LEFT(x, count)`, `RIGHT(x, count)`
  - 좌측/우측에서 count 문자만큼만 추출
- `STROPS()`
- `INSTR()`
  - instr() strpos()도 문자열의 위치를 찾는 역할은 동일하다. 단, instr()의 경우 검색 시작 위치나 (여러 개가 있을 경우) 몇 번째 문자열의 위치를 조회할 것인지에 대해 조건을 조금 더 세세 하게 지정할 수 있다.
- `REPLACE()`
- `LENGTH()`

0330

```
-- 問題１
-- SELECT
--   CONCAT(
--       product_category,
--       ":",
--       product_name
--     ) AS cat_product,

--   cost
-- FROM `sample.products`
-- ORDER BY cost DESC
-- LIMIT 3;

-- 問題２
-- SELECT
--   CONCAT(product_category, "-", product_name) AS cat_produc
--   LENGTH(CONCAT(product_category, "-", product_name)) AS le
-- FROM `sample.products`
-- ORDER BY length DESC
-- LIMIT 3;

-- 問題３ - A（LIKE文）
SELECT
  user_id,
  name
FROM
  `sample.customers`
WHERE
  name LIKE "_____ %"
LIMIT 5;

-- 問題３ - B（文字列の関数）
-- SELECT
--   user_id,
--   name
-- FROM
--   `sample.customers`
-- WHERE

  -- 空白込みの文字数が7で、4文字目が空白であること（注意：空白が複数入力
```

```
  -- LENGTH(SUBSTR(name, 1, STRPOS(name, " ")-1)) = 3 AND LENG
  LENGTH(name) = 7 AND STRPOS(name, " ") = 4
-- LIMIT 100;
```

정규 표현을 이용하는 함수

- `regexp_contains(문자열, 정규표현식)`
  - 정규표현식으로 지정한 문자열 패턴이 포함되어 있는 지 검사.
  - 포함되어 있으면 TRUE, 그렇지 않으면 FALSE를 반환.
- `regexp_extract(대상 문자열, 정규표현, [검색시작위치], [출현횟수])`
  - 정규표현식에 해당하는 문자열 패턴이 2개 있는 경우, 출현횟수를 지정하지 않으면 처음 매치된 문자열을 가져옴.
  - 검색 시작위치를 지정할 경우, 지정한 위치 이후부터 검색한다.
  - 검색시작위치, 출현횟수는 기본값 1

날짜/시각을 다루는 함수

현재 날짜/시각

- `CURRENT_DATE([TIMEZONE])` (DATE형으로 현재 날짜를 반환)
- `CURRENT_DATETIME([TIMEZONE])` (DATETIME형으로 현재 날짜와 시각을 반환)
- `CURRENT_TIME([TIMEZONE])` (TIME형으로 현재 시각을 반환)
- `CURRENT_TIMESTAMP([TIMEZONE])` (TIMESTAMP형으로 현재 TIMESTAMP를 반환)
  - Timezone은 "+9" 혹은 "Asia/Seoul"과 같이 입력할 수 있다. 기본값은 UTC.

날짜/시각 연산

- `DATE_ADD(DATE, INTERVAL)`
- `DATETIME_ADD(DATETIME, INTERVAL)`
- `DATE_DIFF(DATE_NEW, DATE_OLD, UNIT)`
  - UNIT: `year`, `quarter`, `month`, `week`, `day`
- `DATETIME_DIFF(DATETIME_NEW, DATETIME_OLD, UNIT)`

- `DATE_TRUNC(DATE, UNIT)`
- `DATETIME_TRUNC(DATETIME, UNIT)`

정보 추출

- `EXTRACT(<VALUE> FROM <UNIT>)`
  - UNIT: `YEAR`, `QUARTER`, `MONTH`, `WEEK`, `DAYOFYEAR`, `DAY`, `DAYOFWEEK`
    - WEEK: 그 해의 주차
    - DAYOFWEEK: 일요일부터 토요일까지 1-7
- `FORMAT_DATE(FORMAT, DATE)`
- `FORMAT_DATETIME(FORMAT, DATETIME)`
  - `#TODO` 키워드 정리