국가별 기본정보 비교/분석
- gdp, 산업, phone 량 중심으로 -

개요

본 자료는 세계 각국의 전반적인 데이터에서 제기될 수 있는 간단한 질문들을 분석한 것이다.

데이터 정보

Main_Table Data: https://www.kaggle.com/fernandol/countries-of-the-world

- 세계 각국 주요 정보를 제공 (국가,지역,인구, GDP, 산업, 출생/사망률 등등)

참고자료:

Sub_Table Data : https://www.kaggle.com/sovannt/world-bank-youth-unemployment

- 세계 각국 2010-2014 년 청(소)년 실업률 데이터 제공

테이블 생성

```
CREATE TABLE COUNTRIES
  (    "COUNTRY" VARCHAR2(30 BYTE) NOT NULL ENABLE,
       "REGION" VARCHAR2(50 BYTE),
       "POPULATION" NUMBER(10,0),
       "AREA" NUMBER(10,0),
       "POP_DENSITY" NUMBER(10,5),
       "COASTLINE" NUMBER(10,5),
       "NET_MIGRATION" NUMBER(10,5),
       "INFANT_MORTALITY" NUMBER(10,0),
       "GDP" NUMBER(10,0),
       "LITERACY" NUMBER(10,5),
       "PHONES" NUMBER(10,5),
       "ARABLE" NUMBER(10,5),
       "CROPS" NUMBER(10,5),
       "OTHER" NUMBER(10,5),
```

```
    "CLIMATE" NUMBER(10,0),
    "BIRTHRATE" NUMBER(10,5),
    "DEATHRATE" NUMBER(10,5),
    "AGRICULTURE" NUMBER(10,5),
    "INDUSTRY" NUMBER(10,5),
    "SERVICE" NUMBER(10,5),
    "Y2011" NUMBER(20,10)
);
```

Select * from countries;

	COUNTRY	REGION	POPULATION	AREA	POP_DENSITY	COASTLINE	NET_MIGRATION	INFANT_MORTALITY	GDP	LITERACY	PHONES	ARABLE	CROPS	OTHER	CLIMATE	BIRTHRATE	DEATH
1	Afghanistar	ASIA (EX.	31056997	647500	48	0	23.06	163	700	36	3.2	12.13	0.22	87.65	1	46.6	
2	Albania	EASTERN I	3581655	28748	124.6	1.26	4.93	22	4500	86.5	71.2	21.09	4.42	74.49	3	15.11	
3	Algeria	NORTHERI	32930091	238174	13.8	0.04	0.39	31	6000	70	78.1	3.22	0.25	96.53	1	17.14	
4	American S.	OCEANIA	57794	199	290.4	58.29	20.71	9	8000	97	259.5	10	15	75	2	22.46	
5	Andorra	WESTERN	71201	468	152.1	0	6.6	4	19000	100	497.2	2.22	0	97.78	3	8.71	
6	Angola	SUB-SAHA	12127071	124670	9.7	0.13	0	191	1900	42	7.8	2.41	0.24	97.35	(null)	45.11	
7	Anguilla	LATIN AMI	13477	102	132.1	59.8	10.76	21	8600	95	460	0	0	100	2	14.17	
8	Antigua & E	LATIN AMI	69108	443	156	34.54	6.15	19	11000	89	549.9	18.18	4.55	77.27	2	16.93	
9	Argentina	LATIN AMI	39921833	276689	14.4	0.18	0.61	15	11200	97.1	220.4	12.31	0.48	87.21	3	16.73	
10	Armenia	C.W. OF I	2976372	29800	99.9	0	6.47	23	3500	98.6	195.7	17.55	2.3	80.15	4	12.07	
11	Aruba	LATIN AMI	71891	193	372.5	35.49	0	6	28000	97	516.1	10.53	0	89.47	2	11.03	
12	Australia	OCEANIA	20264082	768685	2.6	0.34	3.98	5	29000	100	565.5	6.55	0.04	93.41	1	12.14	
13	Austria	WESTERN	8192880	83870	97.7	0	2	5	30000	98	452.2	16.91	0.86	82.23	3	8.74	
14	Azerbaijan	C.W. OF I	7961619	86600	91.9	0	4.9	82	3400	97	137.1	19.63	2.71	77.66	1	20.74	
15	Bahamas, T	LATIN AMI	303770	13940	21.8	25.41	2.2	25	16700	95.6	460.6	0.8	0.4	98.8	2	17.57	
16	Bahrain	NEAR EAS	698585	665	1050.5	24.21	1.05	17	16900	89.1	281.3	2.82	5.63	91.55	1	17.8	

질문

1. Gdp per capita(이하 'gdp')상위 50 개국들의 산업구조는 어떤 특징을 가지고 있는가?

```
SELECT * FROM(
SELECT country, DENSE_RANK() OVER(ORDER BY gdp desc)rnk, agriculture, industry, service
FROM COUNTRIES_1
WHERE gdp IS NOT NULL
AND agriculture IS NOT null
AND industry IS NOT null
and service IS NOT null)
WHERE rnk<=50;
```

	COUNTRY	RNK	AGRICULTURE	INDUSTRY	SERVICE
1	Luxembourg	1	0.01	0.13	0.86
2	United States	2	0.01	0.204	0.787
3	Norway	2	0.021	0.415	0.564
4	Bermuda	3	0.01	0.1	0.89
5	Cayman Island	4	0.014	0.032	0.954
6	Switzerland	5	0.015	0.34	0.645
7	Denmark	6	0.018	0.246	0.735
8	Iceland	7	0.086	0.15	0.765
9	Austria	8	0.018	0.304	0.678
10	Canada	9	0.022	0.294	0.684
11	Ireland	10	0.05	0.46	0.49
12	Belgium	11	0.01	0.24	0.749
13	Australia	12	0.038	0.262	0.7
14	Hong Kong	13	0.001	0.092	0.906
15	Netherlands	14	0.021	0.244	0.736
16	Japan	15	0.017	0.258	0.725
17	Aruba	16	0.004	0.333	0.663
18	United Kingdom	17	0.005	0.237	0.758
19	France	18	0.022	0.214	0.764
20	Germany	18	0.009	0.296	0.695
21	Finland	19	0.028	0.295	0.676
22	Sweden	20	0.011	0.282	0.707

```
SELECT * FROM(

SELECT country, DENSE_RANK() OVER(ORDER BY gdp asc)rnk, agriculture, industry, service

FROM COUNTRIES_1
WHERE gdp IS NOT NULL

AND agriculture IS NOT null

AND industry IS NOT null

and service IS NOT null)

WHERE rnk<=50;
```

	COUNTRY	RNK	AGRICULTURE	INDUSTRY	SERVICE
1	Somalia	1	0.65	0.1	0.25
2	Sierra Leone	1	0.49	0.31	0.21
3	East Timor	1	0.085	0.231	0.684
4	Tanzania	2	0.432	0.172	0.396
5	Malawi	2	0.342	0.158	0.499
6	Gaza Strip	2	0.03	0.283	0.687
7	Burundi	2	0.463	0.203	0.334
8	Afghanistan	3	0.38	0.24	0.38
9	Ethiopia	3	0.475	0.099	0.426
10	Eritrea	3	0.102	0.254	0.643
11	Congo, Repub	3	0.062	0.57	0.369
12	Congo, Dem. I	3	0.55	0.11	0.34
13	Comoros	3	0.4	0.04	0.56
14	West Bank	4	0.09	0.28	0.63
15	Niger	4	0.39	0.17	0.44
16	Madagascar	4	0.276	0.165	0.559
17	Kiribati	4	0.089	0.242	0.668
18	Guinea-Bissau	4	0.62	0.12	0.26
19	Zambia	4	0.22	0.29	0.489
20	Yemen	4	0.135	0.472	0.393
21	Mali	5	0.45	0.17	0.38
22	Nigeria	5	0.269	0.487	0.244

1 번 문제에 대한 결론 :

1 인당 gdp 가 높은 국가는 농업, 공업, 서비스업 가운데 서비스산업의 비중이 높은 모습을 볼 수 있다. 반대로 하위 gdp 에서는 service 산업이 상대적으로 낮은 모습을 볼 수 있다.

질문

2. 그렇다면 service 산업이 발달한 국가가 phone 보유율이 높은가?

- 분석방법 : service 랭크 상/하위 50 위를 출력해서 각 산업별(service, industry, agriculture)상/하위 50 위

  공통된 국가의 개수를 count 한다. 어떤 산업의 상위 50 개의 수와 service 상위 50 개에 공통된 국가(intersect)가
  가정도 성립한다면 그 산업은 다른 산업에 비해 상대적 phones 와 비례한다고 볼 수 있다.

■ 서비스업

```
SELECT COUNT(*) FROM(
SELECT country FROM(
SELECT country, service, RANK() OVER(ORDER BY service desc)rnk
FROM COUNTRIES)
WHERE rnk<=50
INTERSECT
SELECT country FROM(
SELECT country, phones, RANK() OVER(ORDER BY phones DESC)rnk
FROM COUNTRIES
WHERE phones IS NOT null)
WHERE rnk<=50);
```

	COUNT(*)
1	29

```
SELECT COUNT(*) FROM(
SELECT country FROM(
SELECT country, service, RANK() OVER(ORDER BY service asc)rnk
FROM COUNTRIES)
WHERE rnk<=50
INTERSECT
SELECT country FROM(
SELECT country, phones, RANK() OVER(ORDER BY phones asc)rnk
FROM COUNTRIES
WHERE phones IS NOT null)
WHERE rnk<=50);
```

	COUNT(*)
1	30

- ■ 공업

```
SELECT COUNT(*) FROM(
```

```
SELECT country FROM(
SELECT country, industry, RANK() OVER(ORDER BY industry desc)rnk
FROM COUNTRIES)
WHERE rnk<=50
INTERSECT
SELECT country FROM(
SELECT country, phones, RANK() OVER(ORDER BY phones desc)rnk
FROM COUNTRIES
WHERE phones IS NOT null)
WHERE rnk<=50);
```

	COUNT(*)
1	11

```
SELECT country FROM(
SELECT country, industry, RANK() OVER(ORDER BY industry asc)rnk
FROM COUNTRIES)
WHERE rnk<=50
INTERSECT
SELECT country FROM(
SELECT country, phones, RANK() OVER(ORDER BY phones asc)rnk
FROM COUNTRIES
WHERE phones IS NOT null)
WHERE rnk<=50);
```

	COUNT(*)
1	18

- ■ 농업

```
SELECT COUNT(*) FROM(
SELECT country FROM(
SELECT country, agriculture, RANK() OVER(ORDER BY agriculture desc)rnk
FROM COUNTRIES)
```

```
WHERE rnk<=50
INTERSECT
SELECT country FROM(
SELECT country, phones, RANK() OVER(ORDER BY phones desc)rnk
FROM COUNTRIES
WHERE phones IS NOT null)
WHERE rnk<=50);
```

	COUNT(*)
1	6

```
SELECT COUNT(*) FROM(
SELECT country FROM(
SELECT country, agriculture, RANK() OVER(ORDER BY agriculture asc)rnk
FROM COUNTRIES)
WHERE rnk<=50
INTERSECT
SELECT country FROM(
SELECT country, phones, RANK() OVER(ORDER BY phones asc)rnk
FROM COUNTRIES
WHERE phones IS NOT null)
WHERE rnk<=50);
```

	COUNT(*)
1	1

종합

	PHONES_HIGH	PHONES_LOW	COMMON_RESULTS
SERVICE	29	30	59
INDUSTRY	11	18	29
AGRICULTURE	6	1	7

2 번 문제에 대한 결론:

다른 산업에 비해 서비스산업의 발달정도는 휴대폰 사용률과 비례한다.

질문

3. 가장 생활 환경이 열악한 국가는?

- 기준 : 이주율(순이동율), 영아사망률가 높고, gdp, literacy(문해력),휴대폰사용율이 낮은 국가(1 부터 50 중 등급이 낮을수록 열악하다.)

- 국가별 등급을 한눈에 보기 쉽도록 테이블을 생성.

- 컬럼별 50 등급 나누기

-------- view 생성

```
CREATE OR replace VIEW v1
as
SELECT country, TRIM(LOWER(country)) AS net_mig,
ntile(50) OVER(ORDER BY net_migration desc)grade
FROM COUNTRIES
WHERE net_migration IS NOT NULL;
```

- Create or replace view 는 create view 와 달리 기존에 같은 이름의 뷰가 있으면 덮어쓰는 특성이 있는데 만약 같은 이름의 뷰가 있을 경우 코드를 다시 수행해야 하는 번거로움이 있어서 사용하였다.
- Country 컬럼의 데이터 는 'Angola ' 처럼 initcap 에 마지막에 space 가 있는 형태여서 trim 과 lower 을 써서 'angola' 와 같이 쓰거나 검색하기 쉽도록 바꾸었다.

```
Select * from v1
```

	COUNTRY	NET_MIG	GRADE
1	Afghanistan	afghanistan	1
2	Micronesia, Fed. St.	micronesia, fed. st.	1
3	American Samoa	american samoa	1
4	Cayman Islands	cayman islands	1
5	Qatar	qatar	1
6	Kuwait	kuwait	2
7	Grenada	grenada	2
8	Dominica	dominica	2
9	Cape Verde	cape verde	2
10	Samoa	samoa	2
11	Turks & Caicos Is	turks & caicos is	3
12	Singapore	singapore	3
13	San Marino	san marino	3
14	Trinidad & Tobago	trinidad & tobago	3
15	Anguilla	anguilla	3
16	British Virgin Is.	british virgin is.	4
17	N. Mariana Islands	n. mariana islands	4
18	Luxembourg	luxembourg	4
19	Virgin Islands	virgin islands	4

```
CREATE OR replace VIEW v2

as

SELECT country, TRIM(LOWER(country)) AS inf_mort,

ntile(50) OVER(ORDER BY infant_mortality desc)grade

FROM COUNTRIES

WHERE infant_mortality IS NOT NULL;


CREATE OR replace VIEW v3

as

SELECT country, TRIM(LOWER(country)) AS gdp,

ntile(50) OVER(ORDER BY gdp asc)grade

FROM COUNTRIES

WHERE gdp IS NOT NULL;
```

```
CREATE OR replace VIEW v4
as
SELECT country, TRIM(LOWER(country)) AS literacy,
ntile(50) OVER(ORDER BY literacy asc)grade
FROM COUNTRIES
WHERE literacy IS NOT NULL;

CREATE OR replace VIEW v5
as
SELECT country, TRIM(LOWER(country)) AS phones,
ntile(50) OVER(ORDER BY phones asc)grade
FROM COUNTRIES
WHERE phones IS NOT NULL;

---------- 해당 컬럼의 등급과 국가를 하나의 컬럼으로
CREATE OR REPLACE VIEW v_11
as
SELECT country, net_mig||'('||grade||')' AS net_mig
FROM v1;
SELECT * FROM v_11;
```

- Select 문의 컬럼 중 country 는 view 를 조인할 때 사용될 country 이고,
  컬럼 net_mig 는 이후에 최종적으로 컬럼별 데이터 수치 대신 들어갈 국가와 등급의 데이터를 제공한다.
  V_11 를 select 하면 아래와 같다.

	COUNTRY	NET_MIG
1	Afghanistan	afghanistan(1)
2	Micronesia, Fed. St.	micronesia, fed. st.(1)
3	American Samoa	american samoa(1)
4	Cayman Islands	cayman islands(1)
5	Qatar	qatar(1)
6	Kuwait	kuwait(2)
7	Grenada	grenada(2)
8	Dominica	dominica(2)
9	Cape Verde	cape verde(2)
10	Samoa	samoa(2)
11	Turks & Caicos Is	turks & caicos is(3)
12	Singapore	singapore(3)
13	San Marino	san marino(3)
14	Trinidad & Tobago	trinidad & tobago(3)
15	Anguilla	anguilla(3)
16	British Virgin Is.	british virgin is.(4)
17	N. Mariana Islands	n. mariana islands(4)
18	Luxembourg	luxembourg(4)
19	Virgin Islands	virgin islands(4)

```
 CREATE OR REPLACE VIEW v_21

as

SELECT country, inf_mort||'('||grade||')' AS inf_mort

FROM v2;


CREATE OR REPLACE VIEW v_31

as

SELECT country, gdp||'('||grade||')' AS gdp

FROM v3;


CREATE OR REPLACE VIEW v_41

as

SELECT country, literacy||'('||grade||')' AS literacy
```

```
FROM v4;

CREATE OR REPLACE VIEW v_51

as

SELECT country, phones||'('||grade||')' AS phones

FROM v5;


SELECT * FROM v_11;

SELECT country

FROM v_21

MINUS

SELECT country

FROM v_11;


SELECT country

FROM v_11

MINUS

SELECT country

FROM v_21;
```

- view 들을 Join 하기 전 위에서 수행했던 is not null 에 의해 제거된 row 들을 각각 비교하는 작업이다.

```
CREATE VIEW v_12

as

SELECT v1.country, v1.net_mig, v2.inf_mort

FROM v_11 v1, v_21 v2

WHERE v1.country=v2.country;
```

- country 데이터개수가 같아서 equi join.

```
Select * from V_12
```

▣	COUNTRY	NET_MIG	INF_MORT
1	Angola	angola(35)	angola(1)
2	Afghanistan	afghanistan(1)	afghanistan(1)
3	Sierra Leone	sierra leone(47)	sierra leone(1)
4	Mozambique	mozambique(45)	mozambique(1)
5	Liberia	liberia(42)	liberia(1)
6	Niger	niger(26)	niger(2)
7	Somalia	somalia(7)	somalia(2)
8	Mali	mali(29)	mali(2)
9	Tajikistan	tajikistan(13)	tajikistan(2)
10	Guinea-Bissau	guinea-bissau(2	guinea-bissau(
11	Djibouti	djibouti(38)	djibouti(3)
12	Malawi	malawi(43)	malawi(3)
13	Bhutan	bhutan(36)	bhutan(3)
14	Tanzania	tanzania(17)	tanzania(3)
15	Nigeria	nigeria(31)	nigeria(3)

```
SELECT country
FROM v_31
MINUS
SELECT country
FROM v_12;
```

```
CREATE VIEW v_123
as
SELECT v1.country, v1.net_mig, v1.inf_mort, v2.gdp
FROM v_12 v1, v_31 v2
WHERE v1.country(+)=v2.country;
```

- 두 뷰의 country 를 비교해서 결측된 결과에 따른 outer join.

```
SELECT * FROM v_123;
```

	COUNTRY	NET_MIG	INF_MORT	GDP
1	Angola	angola(35)	angola(1)	angola(12)
2	Afghanistan	afghanistan(1)	afghanistan(1)	afghanistan(2)
3	Sierra Leone	sierra leone(47)	sierra leone(1)	sierra leone(1)
4	Mozambique	mozambique(45)	mozambique(1)	mozambique(6)
5	Liberia	liberia(42)	liberia(1)	liberia(5)
6	Niger	niger(26)	niger(2)	niger(4)
7	Somalia	somalia(7)	somalia(2)	somalia(1)
8	Mali	mali(29)	mali(2)	mali(5)
9	Tajikistan	tajikistan(13)	tajikistan(2)	tajikistan(5)
10	Guinea-Bissau	guinea-bissau(2	guinea-bissau(	guinea-bissau(
11	Djibouti	djibouti(38)	djibouti(3)	djibouti(8)
12	Malawi	malawi(43)	malawi(3)	malawi(2)
13	Bhutan	bhutan(36)	bhutan(3)	bhutan(7)
14	Tanzania	tanzania(17)	tanzania(3)	tanzania(1)
15	Nigeria	nigeria(31)	nigeria(3)	nigeria(5)

```
 SELECT country
FROM v_41
MINUS
SELECT country
FROM v_123;
SELECT country
FROM v_123
MINUS
SELECT country
FROM v_41;


CREATE VIEW v_1234
as
SELECT v1.country, v1.net_mig, v1.inf_mort, v1.gdp, v2.literacy
FROM v_123 v1 FULL outer join v_41 v2
on v1.country=v2.country;
```

- 여기선 두 뷰 모두 서로에게 없는 값이 있어서 full outer join.

```
SELECT * FROM v_1234;


SELECT country
FROM v_51
MINUS
SELECT country
FROM v_1234;


SELECT country
FROM v_1234
MINUS
SELECT country
FROM v_51;


CREATE VIEW v_12345
as
SELECT v1.country, v1.net_mig, v1.inf_mort, v1.gdp, v1.literacy, v2.phones
FROM v_1234 v1 FULL outer join v_51 v2
on v1.country=v2.country;
SELECT * FROM v_12345;


SELECT COUNT(*) FROM v_12345;
```

- 그런데 여기서 v_12345 의 총 데이터의 개수는 230 개로 기존 테이블인 countries 의 국가 개수보다 많게 되었다.
  outer join 을 하면서 서로에게 없던 데이터들이 하나라도 있게 되면
  뷰에 모두 추가하여서 원래의 국가개수보다 많아진 것이다.

	COUNTRY	NET_MIG	INF_MORT	GDP
218	Spain	spain(23)	spain(49)	spain(43)
219	Macau	macau(9)	macau(49)	macau(40)
220	Hong Kong	hong kong(8)	hong kong(49)	hong kong(47)
221	Sweden	sweden(19)	sweden(50)	sweden(45)
222	Japan	japan(41)	japan(50)	japan(46)
223	Iceland	iceland(16)	iceland(50)	iceland(48)
224	Singapore	singapore(3)	singapore(50)	singapore(44)
225	(null)	(null)	(null)	wallis and futur
226	(null)	(null)	(null)	cook islands(22

	COUNTRY	NET_MIG	INF_MORT	GDP	LITERACY
220	Hong Kong	hong kong(8)	hong kong(49)	hong kong(47)	hong kong(27)
221	Sweden	sweden(19)	sweden(50)	sweden(45)	sweden(45)
222	Japan	japan(41)	japan(50)	japan(46)	japan(42)
223	Iceland	iceland(16)	iceland(50)	iceland(48)	iceland(49)
224	Singapore	singapore(3)	singapore(50)	singapore(44)	singapore(25)
225	(null)	(null)	(null)	wallis and futur	(null)
226	(null)	(null)	(null)	cook islands(22	(null)
227	(null)	(null)	(null)	(null)	wallis and futur
228	(null)	(null)	(null)	(null)	cook islands(29

그래서 이 문제는 country 가 null 인 값들을 제거해주면 된다 필요에 따라 쓸 수있다.

이로써 완성된 view 를 보면 기존의 countries 테이블보다 훨씬 보기 쉽다.

```
SELECT * FROM v_12345

WHERE country IS NOT null;


CREATE VIEW t_11

as

SELECT country, net_mig||'('||grade||')' AS net_mig, grade

FROM v1;

SELECT * FROM t_11;
```

	COUNTRY	NET_MIG	GRADE
1	Afghanistan	afghanistan(1)	1
2	Micronesia, Fed. St.	micronesia, fed. st.(1)	1
3	American Samoa	american samoa(1)	1
4	Cayman Islands	cayman islands(1)	1
5	Qatar	qatar(1)	1
6	Kuwait	kuwait(2)	2
7	Grenada	grenada(2)	2
8	Dominica	dominica(2)	2
9	Cape Verde	cape verde(2)	2
10	Samoa	samoa(2)	2
11	Turks & Caicos Is	turks & caicos is(3)	3
12	Singapore	singapore(3)	3
13	San Marino	san marino(3)	3
14	Trinidad & Tobago	trinidad & tobago(3)	3
15	Anguilla	anguilla(3)	3
16	British Virgin Is.	british virgin is.(4)	4
17	N. Mariana Islands	n. mariana islands(4)	4
18	Luxembourg	luxembourg(4)	4
19	Virgin Islands	virgin islands(4)	4

```
CREATE VIEW t_21

as

SELECT country, inf_mort||'('||grade||')' AS inf_mort, grade

FROM v2;

CREATE VIEW t_31

as

SELECT country, gdp||'('||grade||')' AS gdp, grade

FROM v3;


CREATE VIEW t_41

as

SELECT country, literacy||'('||grade||')' AS literacy, grade

FROM v4;
```

```
CREATE VIEW t_51

as

SELECT country, phones||'('||grade||')' AS phones, grade

FROM v5;


CREATE VIEW t_12

AS

SELECT  t1.country, t1.grade AS net_mig, t2.grade AS inf_mort --t1.country, t1.net_mig, t1.grade, t2.inf_mort, t2.grade

FROM t_11 t1, t_21 t2
WHERE t1.country=t2.country;
SELECT * FROM t_12;
```

	COUNTRY	NET_MIG	INF_MORT
1	Angola	35	1
2	Afghanistan	1	1
3	Sierra Leone	47	1
4	Mozambique	45	1
5	Liberia	42	1
6	Niger	26	2
7	Somalia	7	2
8	Mali	29	2
9	Tajikistan	13	2
10	Guinea-Bissau	20	2
11	Djibouti	38	3
12	Malawi	43	3
13	Bhutan	36	3
14	Tanzania	17	3
15	Nigeria	31	3
16	Burkina Faso	37	4
17	Congo, Dem. I	38	4
18	Ethiopia	39	4
19	Chad	33	4

```
CREATE VIEW t_123
```

```
as
SELECT t1.country, t1.net_mig, t1.inf_mort, t2.grade AS gdp
FROM t_12 t1, t_31 t2
WHERE t1.country(+)=t2.country;

SELECT * FROM c_grade;
SELECT * FROM t_123;

CREATE VIEW t_1234
as
SELECT t1.country, t1.net_mig, t1.inf_mort, t1.gdp, t2.grade AS literacy
FROM t_123 t1 FULL outer join t_41 t2
on t1.country=t2.country;

SELECT * FROM t_1234;

CREATE VIEW t_12345
as
SELECT t1.country, t1.net_mig, t1.inf_mort, t1.gdp, t1.literacy, t2.grade AS phones
FROM t_1234 t1 FULL outer join t_51 t2
on t1.country=t2.country;
SELECT * FROM t_12345
WHERE country IS NOT null;
```

	COUNTRY	NET_MIG	INF_MORT	GDP	LITERACY	PHONES
1	Angola	35	1	12	3	5
2	Afghanistan	1	1	2	1	2
3	Sierra Leone	47	1	1	1	4
4	Mozambique	45	1	6	5	3
5	Liberia	42	1	5	7	1
6	Niger	26	2	4	1	1
7	Somalia	7	2	1	2	7
8	Mali	29	2	5	4	4
9	Tajikistan	13	2	5	46	11
10	Guinea-Bissau	20	2	4	3	5
11	Djibouti	38	3	8	10	9

```
SELECT COUNTRY, NET_MIG, INF_MORT, GDP, LITERACY, PHONES,

NET_MIG+ INF_MORT+GDP+LITERACY+PHONES  AS TOTAL

FROM T_12345

ORDER BY TOTAL ASC;
```

	COUNTRY	NET_MIG	INF_MORT	GDP	LITERACY	PHONES	TOTAL
1	Afghanistan	1	1	2	1	2	7
2	Somalia	7	2	1	2	7	19
3	Niger	26	2	4	1	1	34
4	Guinea	13	5	13	1	2	34
5	Guinea-Bissau	20	2	4	3	5	34
6	Tanzania	17	3	1	13	3	37
7	Haiti	12	7	9	6	9	43
8	Mali	29	2	5	4	4	44
9	Chad	33	4	7	4	1	49
10	Gambia, The	20	8	10	2	10	50
11	Pakistan	14	8	13	4	11	50

```
 CREATE TABLE r_grade

as

SELECT COUNTRY, NET_MIG, INF_MORT, GDP, LITERACY, PHONES,

NET_MIG+ INF_MORT+GDP+LITERACY+PHONES  AS TOTAL

FROM T_12345
```

ORDER BY TOTAL ASC;

3 번 문제에 대한 결론 :

가장 살기 열악한 나라는 아프가니스탄!

반대로 가장 환경이 좋은 나라는 이웃나라 일본!

	COUNTRY	NET_MIG	INF_MORT	GDP	LITERACY	PHONES	TOTAL
200	Sweden	sweden(19)	sweden(50)	sweden(45)	sweden(45)	sweden(49)	Sweden (208)
201	Taiwan	taiwan(48)	taiwan(42)	taiwan(43)	taiwan(31)	taiwan(46)	Taiwan (210)
202	Iceland	iceland(16)	iceland(50)	iceland(48)	iceland(49)	iceland(47)	Iceland (210)
203	Aruba	aruba(50)	aruba(41)	aruba(46)	aruba(34)	aruba(44)	Aruba (215)
204	Japan	japan(41)	japan(50)	japan(46)	japan(42)	japan(40)	Japan (219)
205	Mayotte	mayotte(6)	mayotte(11)	mayotte(15)	(null)	mayotte(13)	Mayotte ()
206	Kiribati	kiribati(41)	kiribati(14)	kiribati(4)	(null)	kiribati(13)	Kiribati ()

컬럼 개수가 많거나 데이터의 이름이 길어서 메인 컬럼과 데이터값을 함께 보기 힘들 때는 위와 같이 할 수 있다.

```
CREATE TABLE c_grade

as

SELECT COUNTRY, NET_MIG, INF_MORT, GDP, LITERACY, PHONES,

CONCAT(country||'(', NET_MIG+ INF_MORT+GDP+LITERACY+PHONES)||')' AS TOTAL,

NET_MIG+ INF_MORT+GDP+LITERACY+PHONES AS h_total

FROM T_12345

ORDER BY h_total asc;
```

-- r_grade 에서만 토탈의 순위를 계산할 수 있으므로 여기서 만든 두 컬럼을 c_grade 에 붙인다.

	COUNTRY	NET_MIG	INF_MORT	GDP	LITERACY	PHONES	TOTAL	H_TOTAL
1	Angola	angola(35)	angola(1)	angola(12)	angola(3)	angola(5)	Angola (56)	56
2	Afghanistan	afghanistan(1)	afghanistan(1)	afghanistan(2)	afghanistan(1)	afghanistan(2)	Afghanistan (7)	7
3	Sierra Leone	sierra leone(47)	sierra leone(1)	sierra leone(1)	sierra leone(1)	sierra leone(4)	Sierra Leone (54)	54
4	Mozambique	mozambique(45)	mozambique(1)	mozambique(6)	mozambique(5)	mozambique(3)	Mozambique (60)	60
5	Liberia	liberia(42)	liberia(1)	liberia(5)	liberia(7)	liberia(1)	Liberia (56)	56
6	Niger	niger(26)	niger(2)	niger(4)	niger(1)	niger(1)	Niger (34)	34
7	Somalia	somalia(7)	somalia(2)	somalia(1)	somalia(2)	somalia(7)	Somalia (19)	19
8	Mali	mali(29)	mali(2)	mali(5)	mali(4)	mali(4)	Mali (44)	44
9	Tajikistan	tajikistan(13)	tajikistan(2)	tajikistan(5)	tajikistan(46)	tajikistan(11)	Tajikistan (77)	77
10	Guinea-Bissau	guinea-bissau(2	guinea-bissau(	guinea-bissau(	guinea-bissau(	guinea-bissau(	Guinea-Bissau (34	34
11	Djibouti	djibouti(38)	djibouti(3)	djibouti(8)	djibouti(10)	djibouti(9)	Djibouti (68)	68
12	Malawi	malawi(43)	malawi(3)	malawi(2)	malawi(8)	malawi(5)	Malawi (61)	61
13	Tanzania	tanzania(17)	tanzania(3)	tanzania(1)	tanzania(13)	tanzania(3)	Tanzania (37)	37
14	Nigeria	nigeria(31)	nigeria(3)	nigeria(5)	nigeria(10)	nigeria(6)	Nigeria (55)	55
15	Burkina Faso	burkina faso(37)	burkina faso(4)	burkina faso(6)	burkina faso(1)	burkina faso(4)	Burkina Faso (52)	52
16	Congo, Dem. I	congo, dem. rep	congo, dem. re	congo, dem. re	congo, dem. re	congo, dem. re	Congo, Dem. Rep	54
17	Ethiopia	ethiopia(39)	ethiopia(4)	ethiopia(3)	ethiopia(3)	ethiopia(6)	Ethiopia (55)	55
18	Chad	chad(33)	chad(4)	chad(7)	chad(4)	chad(1)	Chad (49)	49
19	Congo, Reput	congo, repub. o	congo, repub.	congo, repub.	congo, repub.	congo, repub.	Congo, Repub. of	58
20	Cote d'Ivoire	cote d'ivoire(34)	cote d'ivoire(5)	cote d'ivoire(8)	cote d'ivoire(5)	cote d'ivoire(8)	Cote d'Ivoire (60)	60

Sys 권한으로 만든 테이블이 아니라면 ALTER TABLE r_grade_1
SET UNUSED COLUMN h_total; 로 컬럼을 숨기면 되지만 sys 에 의한 테이블은 일일이 select 절에 컬럼을 써 주었다.

Sys 권한의 테이블은 컬럼을 임의로 바꿀 수 없기 때문이다.

	COUNTRY	NET_MIG	INF_MORT	GDP	LITERACY	PHONES	TOTAL
1	Afghanistan	afghanistan(1)	afghanistan(1)	afghanistan(2)	afghanistan(1)	afghanistan(2)	Afghanistan (7)
2	Somalia	somalia(7)	somalia(2)	somalia(1)	somalia(2)	somalia(7)	Somalia (19)
3	Guinea-Bissau	guinea-bissau(20)	guinea-bissau(2)	guinea-bissau(4)	guinea-bissau(3)	guinea-bissau(5)	Guinea-Bissau (34)
4	Guinea	guinea(13)	guinea(5)	guinea(13)	guinea(1)	guinea(2)	Guinea (34)
5	Niger	niger(26)	niger(2)	niger(4)	niger(1)	niger(1)	Niger (34)
6	Tanzania	tanzania(17)	tanzania(3)	tanzania(1)	tanzania(13)	tanzania(3)	Tanzania (37)
7	Haiti	haiti(12)	haiti(7)	haiti(9)	haiti(6)	haiti(9)	Haiti (43)
8	Mali	mali(29)	mali(2)	mali(5)	mali(4)	mali(4)	Mali (44)
9	Chad	chad(33)	chad(4)	chad(7)	chad(4)	chad(1)	Chad (49)
10	Pakistan	pakistan(14)	pakistan(8)	pakistan(13)	pakistan(4)	pakistan(11)	Pakistan (50)

4. 국가를 입력하면 기본등급이 출력되는 pl/sql

```
SET SERVEROUTPUT ON
SET VERIFY OFF

prompt 국가별 생활환경 등급입니다 worst(1) ----- best(50)
ACCEPT p_country PROMPT '국가를 입력하시오'

declare
```

```
v_country r_grade.country%type :=initcap('&p_country ');
v_cnt number(10);
v_exception exception;

begin

select count(*) into v_cnt
from r_grade
where regexp_like(country, v_country);

if v_cnt =0 then
raise v_exception;

else

for r_grade_record in
(select *
from r_grade
where regexp_like(country, v_country)) loop

dbms_output.put_line(r_grade_record.country||' 이주율 :'||r_grade_record.net_mig||'등급 영아사망률 :
'||r_grade_record.inf_mort||'등급 1 인당 gdp : '||
r_grade_record.gdp||'등급 문해력 : '||r_grade_record.literacy||'등급 휴대폰 사용률 : '||r_grade_record.phones||'등급


END LOOP;

end if;

exception
when v_exception then

dbms_output.put_line('해당 국가가 없습니다');

end;
/
```

```
SQL> @ pp1;
국가별 생활환경 등급입니다 worst(1) ----- best(50)
국가를 입력하시오china
China  이주율 :28등급 영아사망률 : 21등급 1인당 gdp : 21등급  문해력 : 23등급
       휴대폰 사용률 : 30등급

PL/SQL 처리가 정상적으로 완료되었습니다.

SQL>
```

결론

1 인당 gdp 가 높은 국가일수록 모습이 다른 산업(농업,공업)에 비해 서버스 산업이 발달한 두드러지게 보였고,

서비스 산업이 발달할수록 휴대폰 사용량도 높았다.

또한 주어진 데이터 상에서 순이동률(국가간), 영아사망률, gdp, 문해력(literacy),

휴대폰사용량에 의한 조건 하에 50 등급을 매겨 5 개의 컬럼에 중요도가 같다고 가정할 때

가장 환경 조건이 열악한 국가는 아프가니스탄이었고,반대로 가장 환경 조건이 좋은 나라는 일본이었다.

참고

만약 혹자가 위의 컬럼에 없는 '세계 청(소)년 실업률'을 위의 테이블과 함께 분석하려고 한다면

merge 또는 join 으로 컬럼을 추가해야 할 것이다.하지만 join 을 위해서 데이터를 정제하는 과정이 필요하다.

예컨대, countries 테이블의 country 컬럼에 우리나라를 지칭하는 데이터 값은'Korea, South '이고,

join 할 테이블에서는 'Korea, Rep.'다.이럴 때 단순히 join 을 하기엔 많은 데이터 손실/중복이 발생할 수 있다.

테이블 생성

```
CREATE TABLE unemployment
(country VARCHAR2(20)
y2011      number(10,5));
```

	COUNTRY	Y2011
1	Madagascar	6.1999998093
2	Maldives	25.5
3	Middle East & North Africa	28.7121316339
4	Mexico	10
5	Middle income	13.42221763
6	Macedonia, FYR	55.4000015259
7	Mali	10.3999996185
8	Malta	14
9	Myanmar	9.3999996185
10	Middle East & North Africa	29.6353922482
11	Montenegro	37.2000007629
12	Mongolia	9
13	Mozambique	40.7000007629
14	Mauritania	46.7000007629
15	Mauritius	21.7000007629
16	Malawi	13.8999996185
17	Malaysia	10
18	North America	17.0869456394
19	Namibia	40.0999984741
20	Niger	7.3000001907
21	Nigeria	13.8000001907
22	Nicaragua	11.6000003815
23	Netherlands	7.5

```
SELECT COUNT(1) FROM(
SELECT TRIM(LOWER(country)) FROM UNEMPLOYMENT
minus
(SELECT * FROM(
SELECT TRIM(LOWER(country)) FROM COUNTRIES
INTERSECT
```

```
SELECT TRIM(LOWER(country)) FROM UNEMPLOYMENT)));
-- trim 과 lower 로 정제되지 않는 데이터 개수

CREATE TABLE UNEMPl
AS
SELECT TRIM(LOWER(country)) AS country, y2011
FROM UNEMPLOYMENT
WHERE TRIM(LOWER(country)) in (
SELECT TRIM(LOWER(country)) FROM COUNTRIES
INTERSECT
SELECT TRIM(LOWER(country)) FROM UNEMPLOYMENT);
-- country 와 2011 년 실업률 데이터 테이블 생성


CREATE TABLE re_data
AS
SELECT TRIM(LOWER(country))AS country, y2011
FROM UNEMPLOYMENT
WHERE TRIM(LOWER(country)) in (
SELECT TRIM(LOWER(country)) FROM UNEMPLOYMENT
minus
SELECT TRIM(LOWER(country)) FROM COUNTRIES);
-- 정제되지 않는 데이터들의 데이터(country, 2011 년 실업률)


ALTER TABLE countries
ADD y2011 number(20,10);
--테이블 2011 년 실업률 데이터 추가

CREATE TABLE unempl_2
```

```
AS
SELECT INITCAP(country)||' 'AS country,y2011
FROM UNEMPLOYMENT;
--Trim 과 lower 로 정제 가능한 데이터 Merge 를 위한 테이블 생성

MERGE INTO COUNTRIES c
USING unempl_2 u
ON (C.country=U.country)
WHEN MATCHED THEN
UPDATE SET c.y2011=u.y2011;
-- merge
SELECT 'delete from re_data '|| 'where '|| 'country' ||' = '||''''||country||''''||';'
from
(SELECT country FROM re_data
WHERE REGEXP_LIKE(country,'(the|world|high|small|income|early|late|fragile|poor|ida|&|area|union|only|pdr|countries|oecd|as
--실업률 테이블 country 컬럼 중 필요없는 단어를 포함하는 국가 제거하는 쿼리

SELECT country, y2011
FROM re_data
WHERE regexp_substr(LOWER(country),'[^|, ]+',1,1)
IN (SELECT regexp_substr(LOWER(country),'[^|, ]+',1,1)ctr_name FROM countries);
--re_data 에서 정규표현식을 사용해서 공통된 글자를 가진 country 데이터 추출

SELECT country
FROM countries
WHERE regexp_substr(LOWER(country),'[^|, ]+',1,1)
IN (SELECT regexp_substr(LOWER(country),'[^|, ]+',1,1) FROM re_data);
--반대로 countries 에서의 country 데이터 형식
begin
for i in
```

```
(SELECT y2011
from re_data
WHERE regexp_substr(LOWER(country),'[^|, ]+',1,1)
IN (SELECT regexp_substr(LOWER(country),'[^|, ]+',1,1)contry FROM countries))
loop
update countries
set y2011=i
where regexp_substr(LOWER(country),'[^|, ]+',1,1)
IN (SELECT regexp_substr(LOWER(country),'[^|, ]+',1,1)contry FROM countries);
end loop;
commit;
end;
/
```

--나머지 re_data 의 국가를 y2011 에 update 하는 Pl/sql 식

결과

	_DENSITY	COASTLINE	NET_MIGRATION	INFANT_MORTALITY	GDP	LITERACY	PHONES	ARABLE	CROPS	OTHER	CLIMATE	BIRTHRATE	DEATHRATE	AGRICULTURE	INDUSTRY	SERVICE	Y2011
2	124.6	1.26	4.93	22	4500	86.5	71.2	21.09	4.42	74.49	3	15.11	5.22	0.232	0.188	0.579	27
3	13.8	0.04	0.39	31	6000	70	78.1	3.22	0.25	96.53	1	17.14	4.61	0.101	0.6	0.298	22.5
4	290.4	58.29	20.71	9	8000	97	259.5	10	15	75	2	22.46	3.27	(null)	(null)	(null)	(null)
5	152.1	0	6.6	4	19000	100	497.2	2.22	0	97.78	3	8.71	6.25	(null)	(null)	(null)	(null)
6	9.7	0.13	0	191	1900	42	7.8	2.41	0.24	97.35	(null)	45.11	24.2	0.096	0.658	0.246	10.6999998093
7	132.1	59.8	10.76	21	8600	95	460	0	0	100	2	14.17	5.34	0.04	0.18	0.78	(null)
8	156	34.54	6.15	19	11000	89	549.9	18.18	4.55	77.27	2	16.93	5.37	0.038	0.22	0.743	(null)
9	14.4	0.18	0.61	15	11200	97.1	220.4	12.31	0.48	87.21	3	16.73	7.55	0.095	0.358	0.547	18.7999992371
10	99.9	0	6.47	23	3500	98.6	195.7	17.55	2.3	80.15	4	12.07	8.23	0.239	0.343	0.418	38.7000007629
11	372.5	35.49	0	6	28000	97	516.1	10.53	0	89.47	2	11.03	6.68	0.004	0.333	0.663	(null)
12	2.6	0.34	3.98	5	29000	100	565.5	6.55	0.04	93.41	1	12.14	7.51	0.038	0.262	0.7	11.3999996185
13	97.7	0	2	5	30000	98	452.2	16.91	0.86	82.23	3	8.74	9.76	0.018	0.304	0.678	8.1999998093
14	91.9	0	4.9	82	3400	97	137.1	19.63	2.71	77.66	1	20.74	9.75	0.141	0.457	0.402	14.5
15	21.8	25.41	2.2	25	16700	95.6	460.6	0.8	0.4	98.8	2	17.57	9.05	0.03	0.07	0.9	27.2000007629
16	1050.5	24.21	1.05	17	16900	89.1	281.3	2.82	5.63	91.55	1	17.8	4.14	0.005	0.387	0.608	11.3999996185
17	1023.4	0.4	0.71	63	1900	43.1	7.3	62.11	3.07	34.82	2	29.8	8.27	0.199	0.198	0.603	8.1999998093
18	649.5	22.51	0.31	13	15700	97.4	481.9	37.21	2.33	60.46	2	12.71	8.67	0.06	0.16	0.78	25.5
19	49.6	0	2.54	13	6100	99.6	319.1	29.55	0.6	69.85	4	11.16	14.02	0.093	0.316	0.591	12.5
20	340	0.22	1.23	5	29100	98	462.6	23.28	0.4	76.32	3	10.38	10.27	0.01	0.24	0.749	18.6000003815
21	12.5	1.68	0	26	4900	94.1	115.7	2.85	1.71	95.44	2	28.84	5.72	0.142	0.152	0.612	24.2999992371
22	69.8	0.11	0	85	1100	40.9	9.7	18.08	2.4	79.52	2	38.85	12.22	0.316	0.138	0.546	, 2