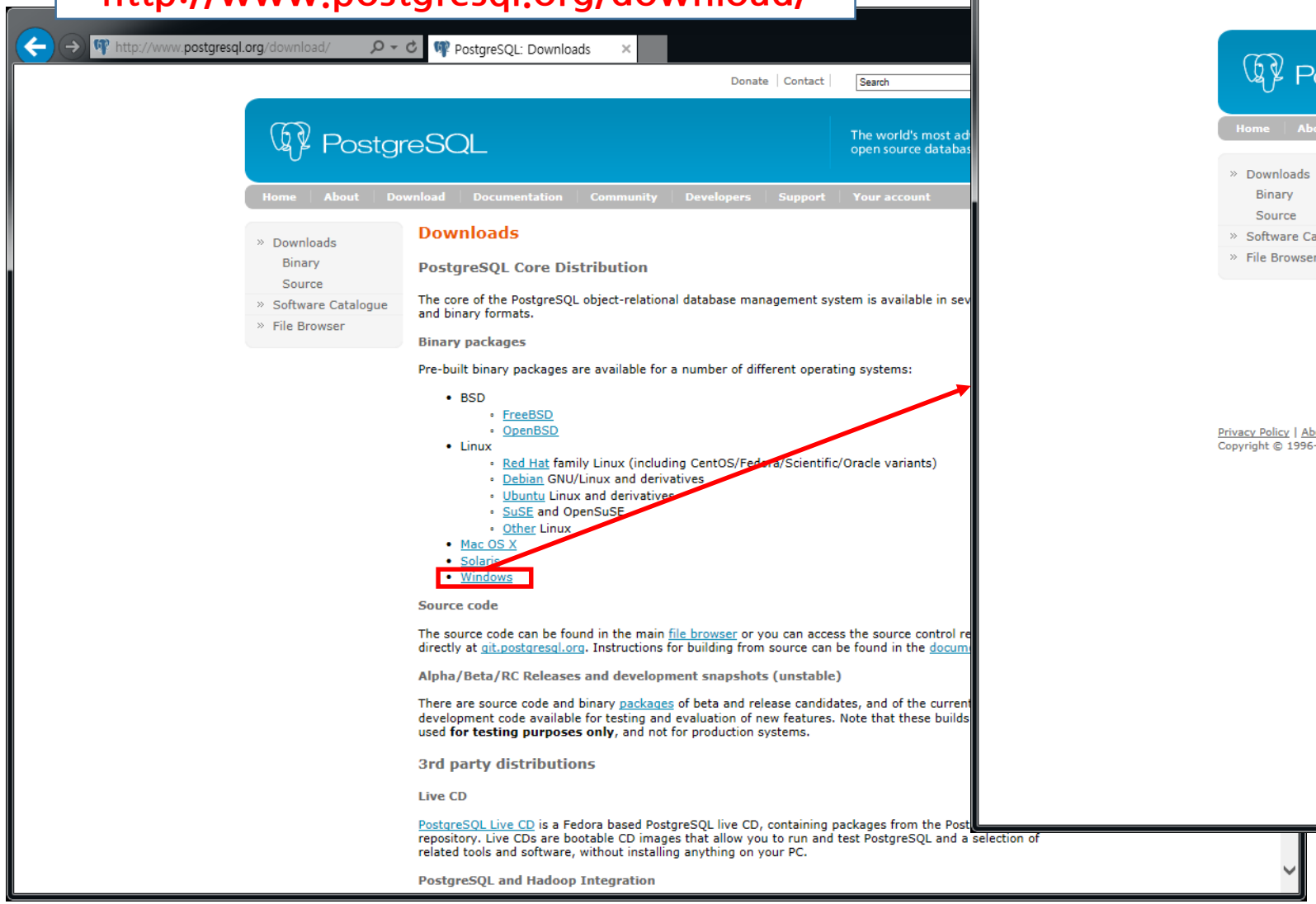
SQL for PostgreSQL

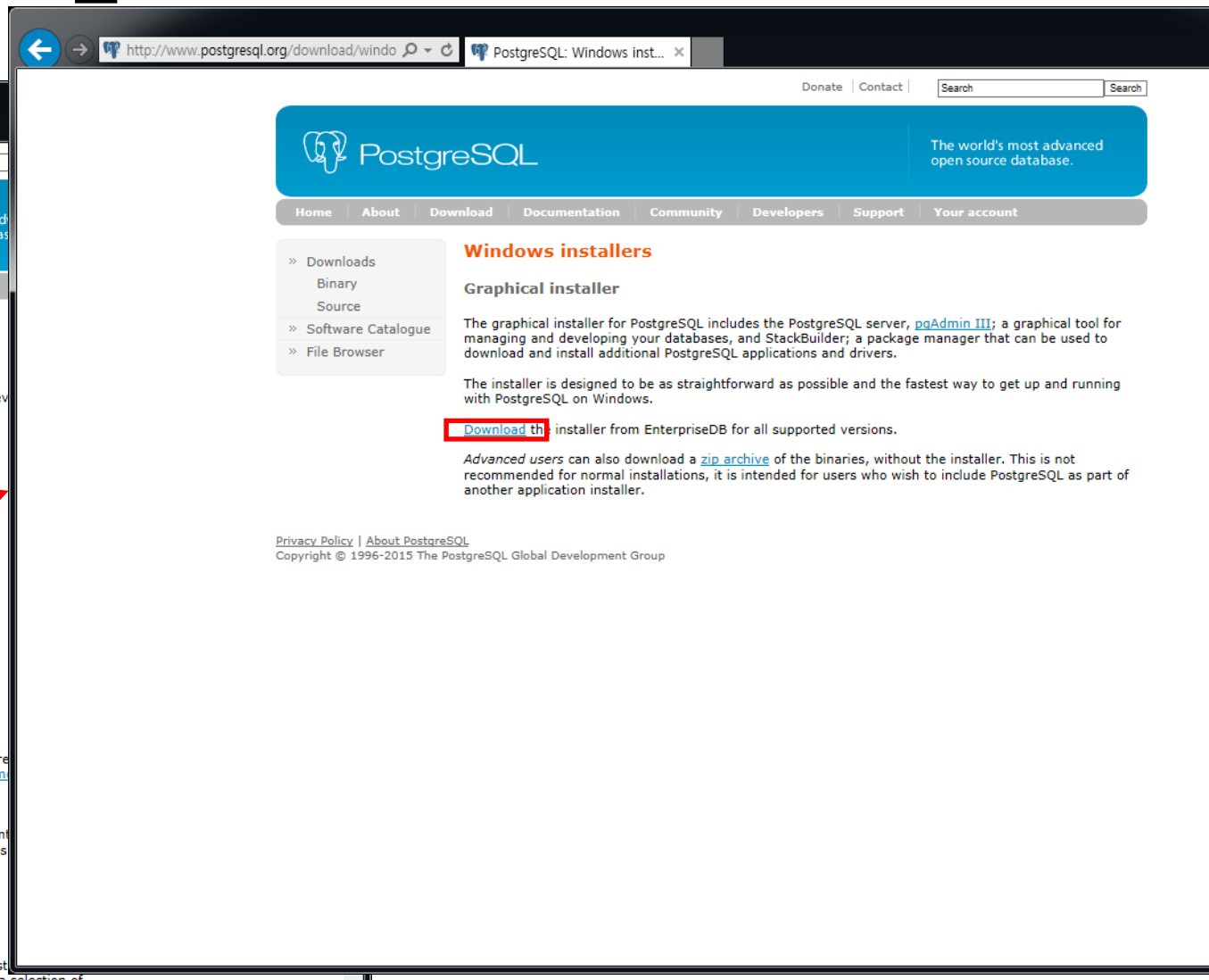
황종필

PostgreSQL Setup

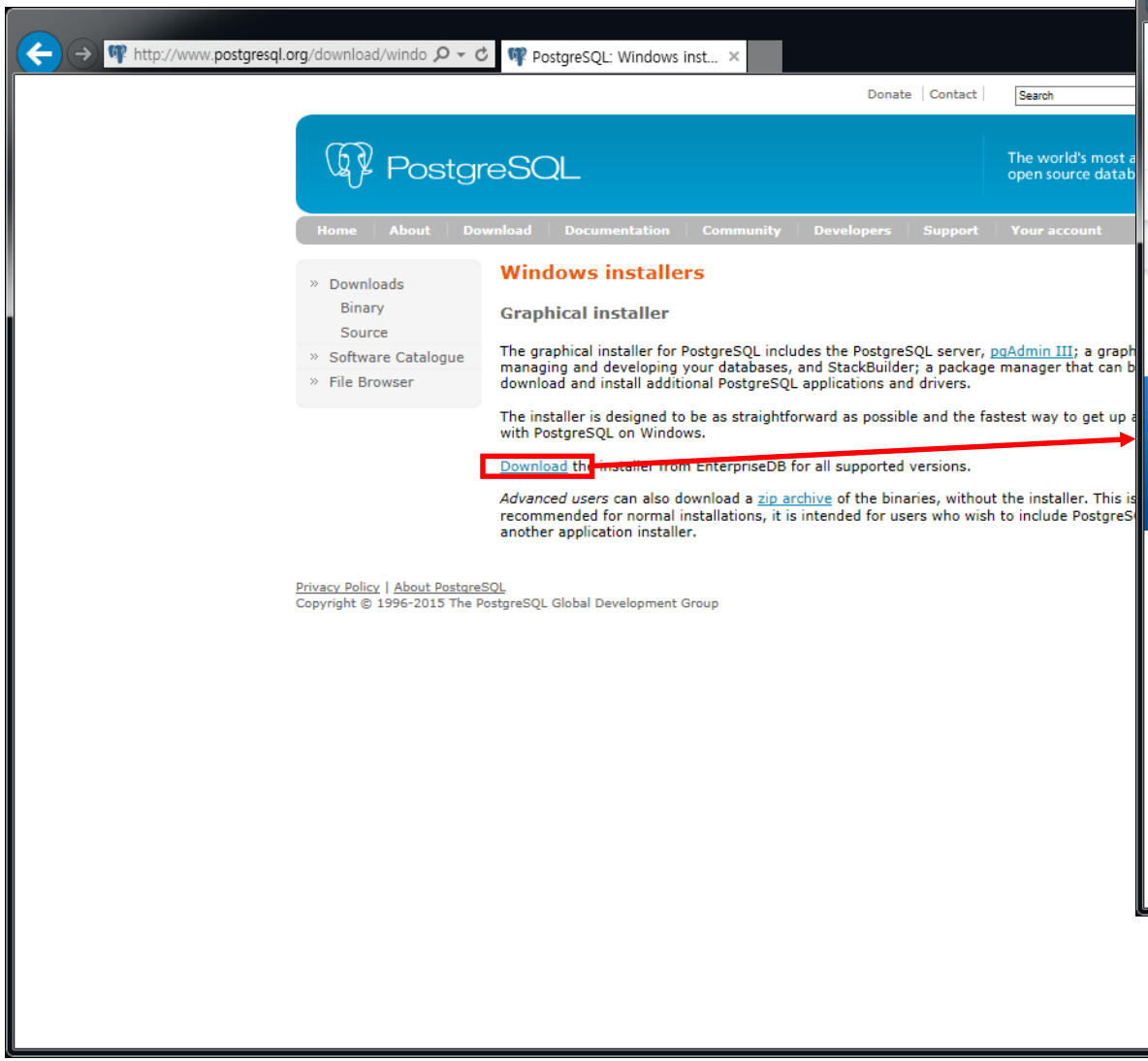
PostgreSQL Setup - 1

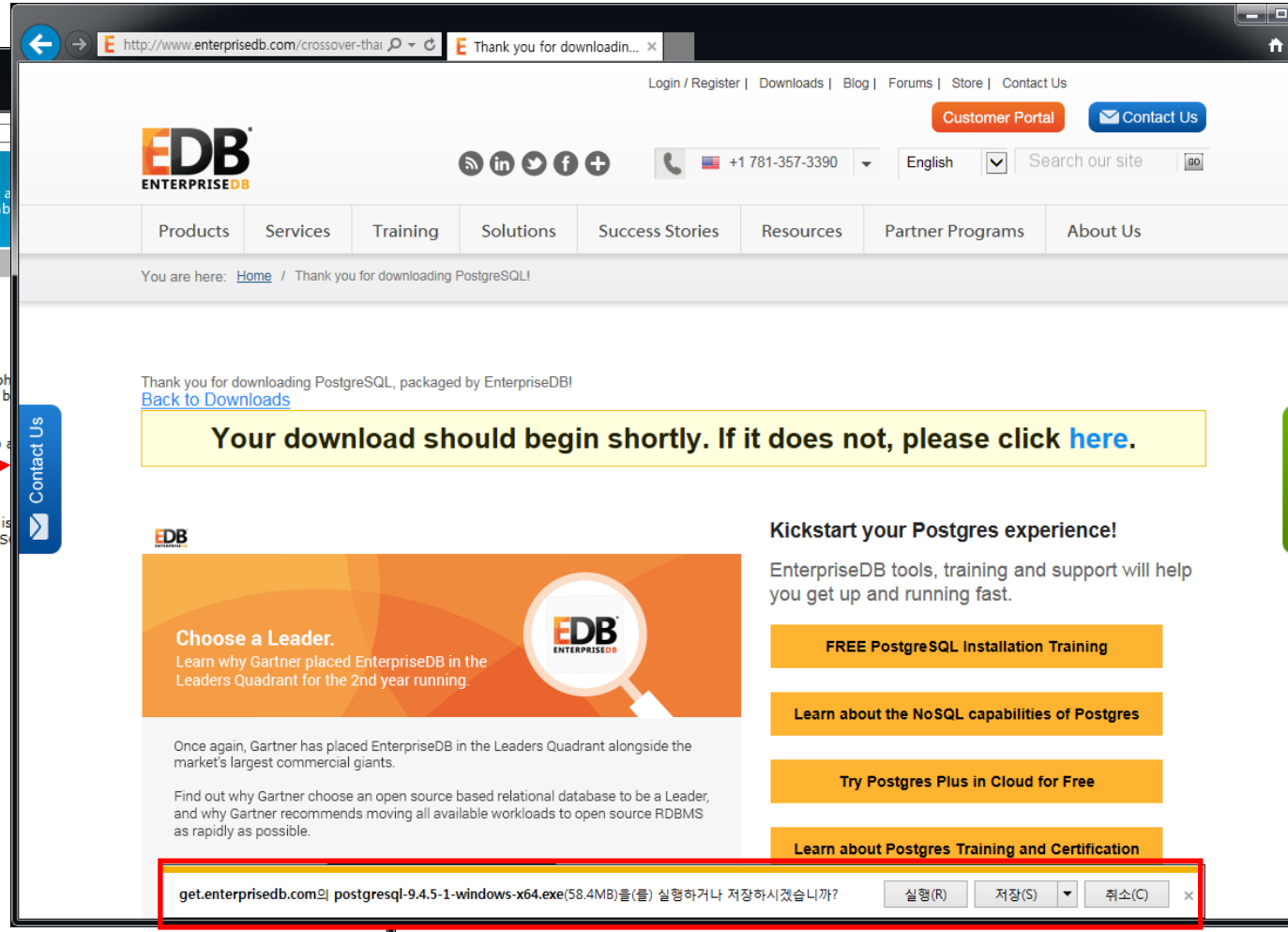
http://www.postgresql.org/download/



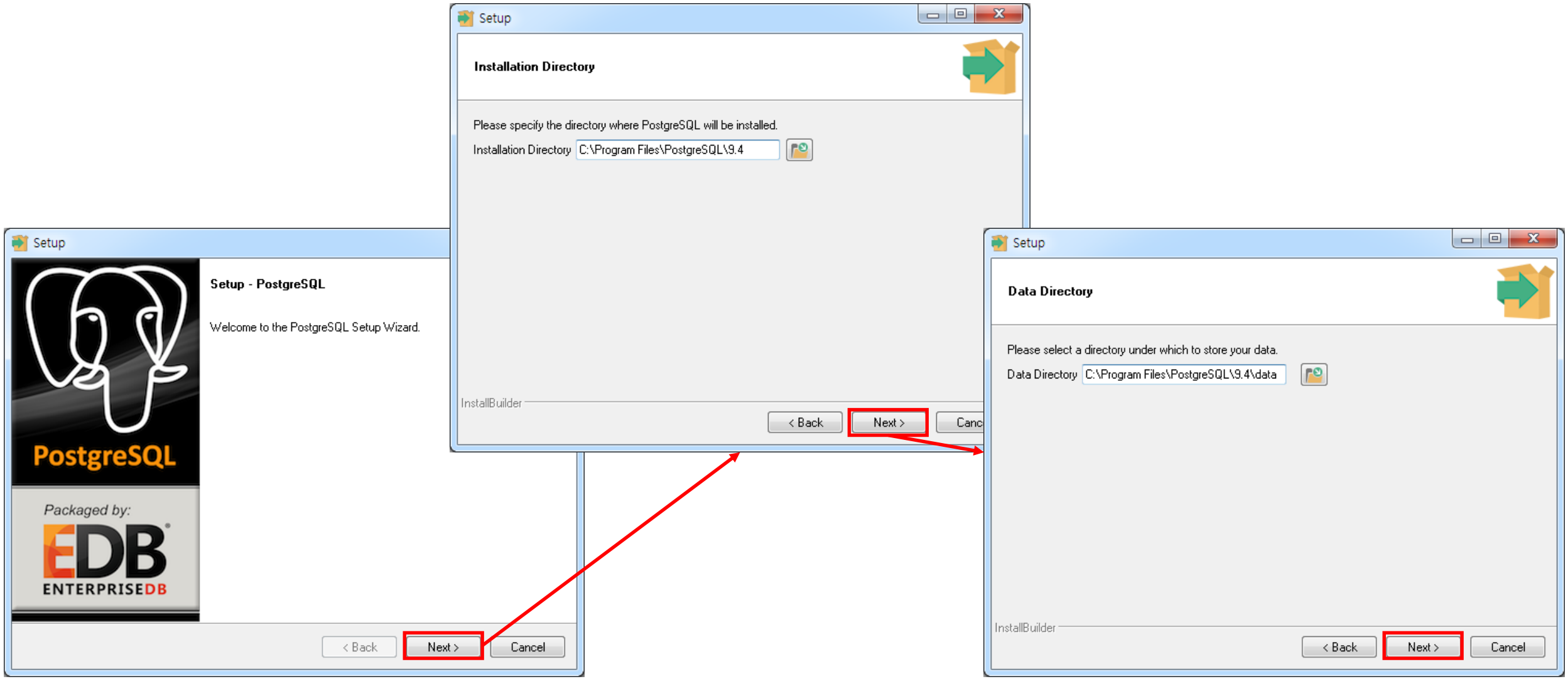


PostgreSQL Setup - 2

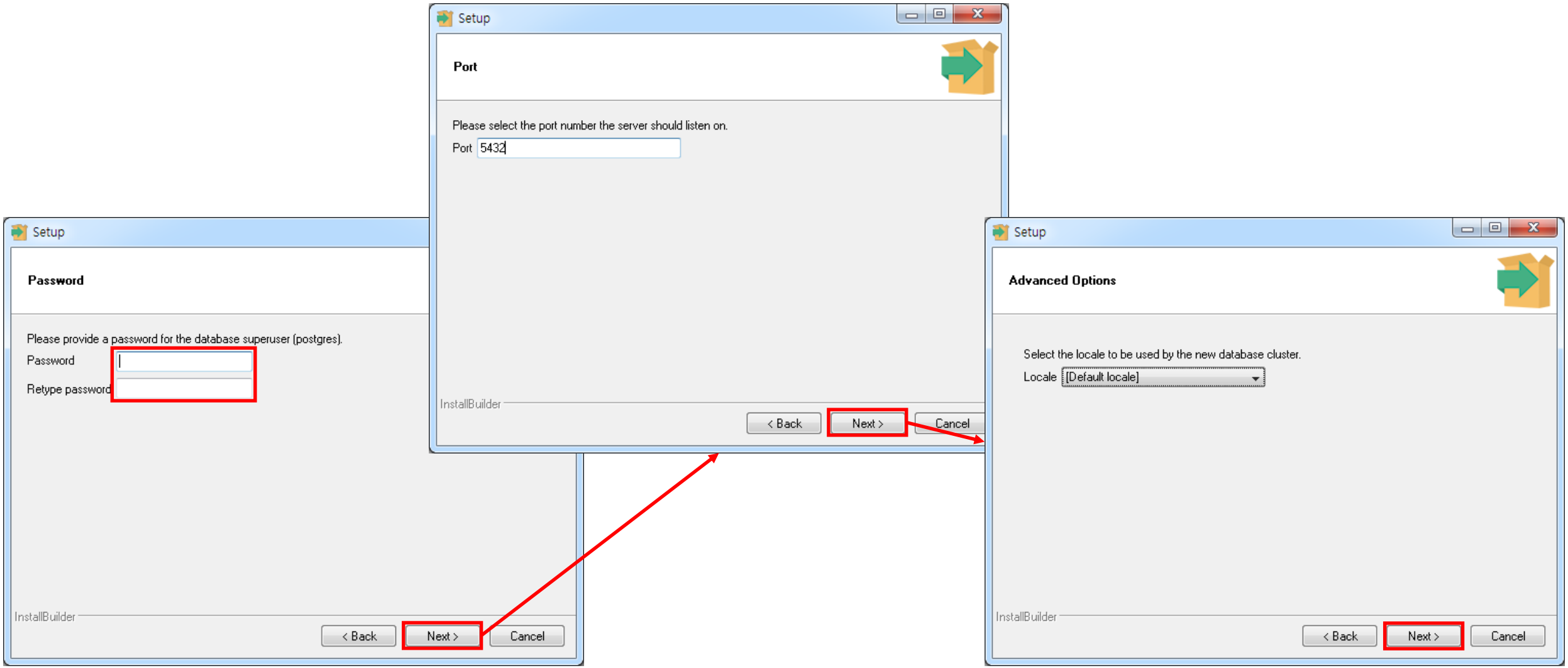




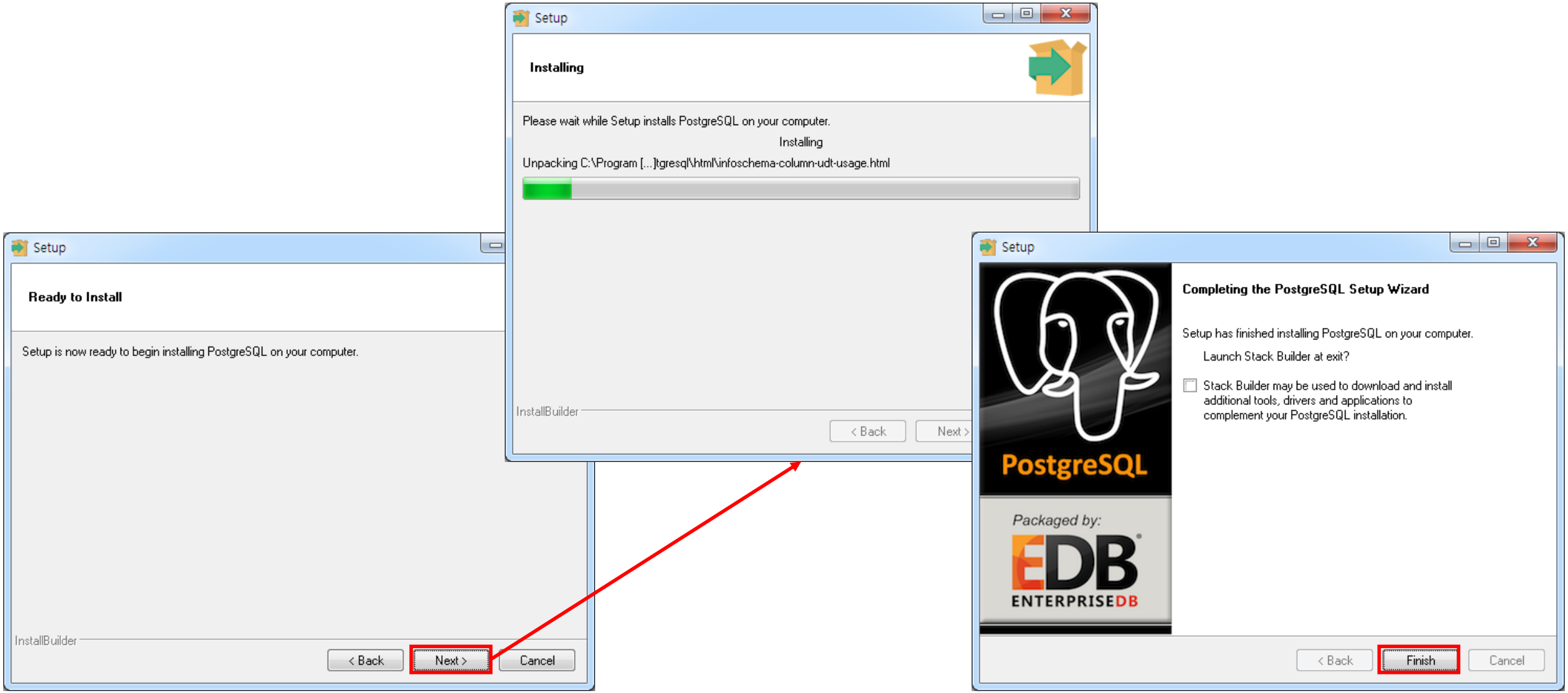
PostgreSQL Setup - 3



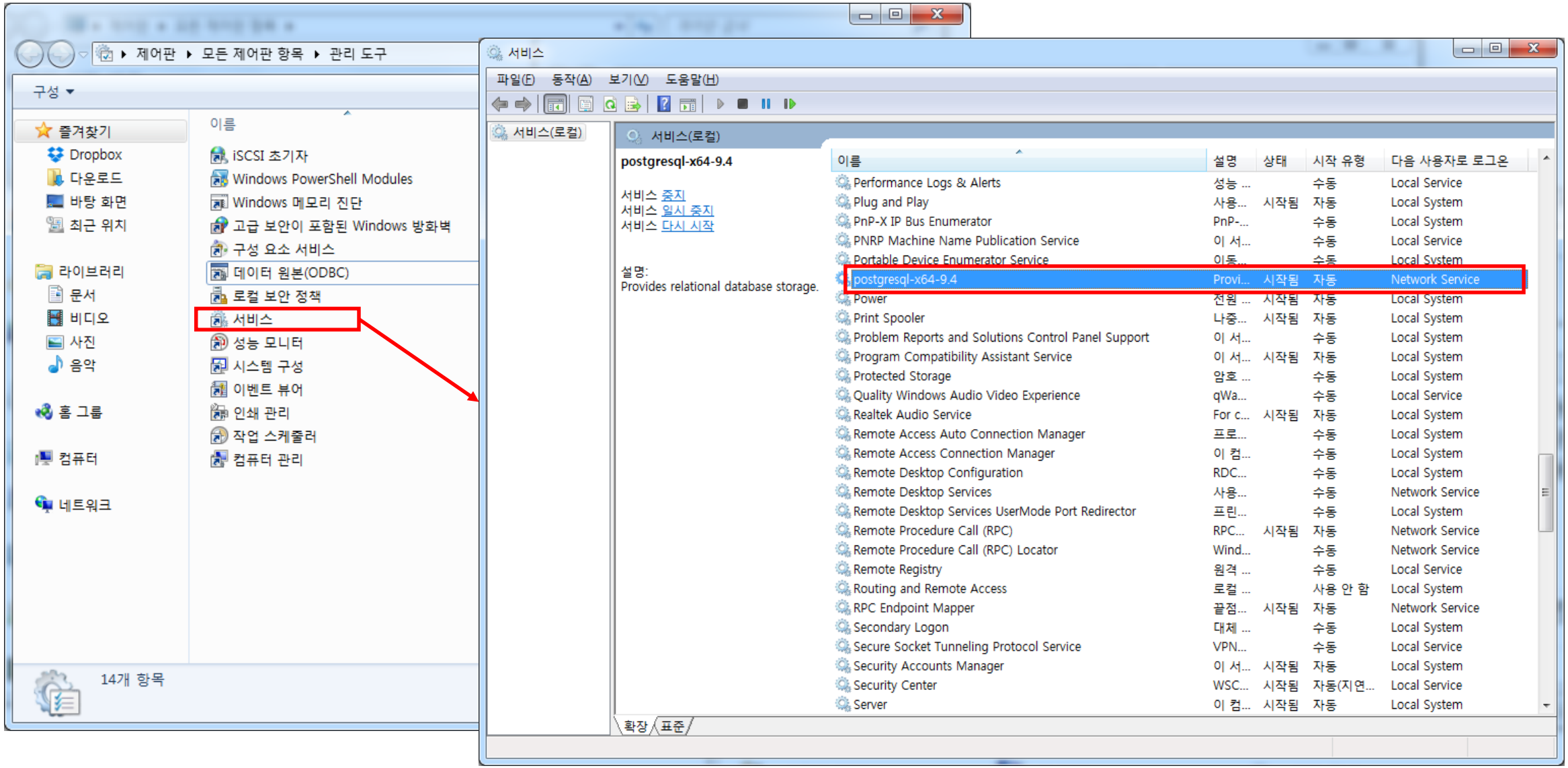
PostgreSQL Setup - 4



PostgreSQL Setup - 5

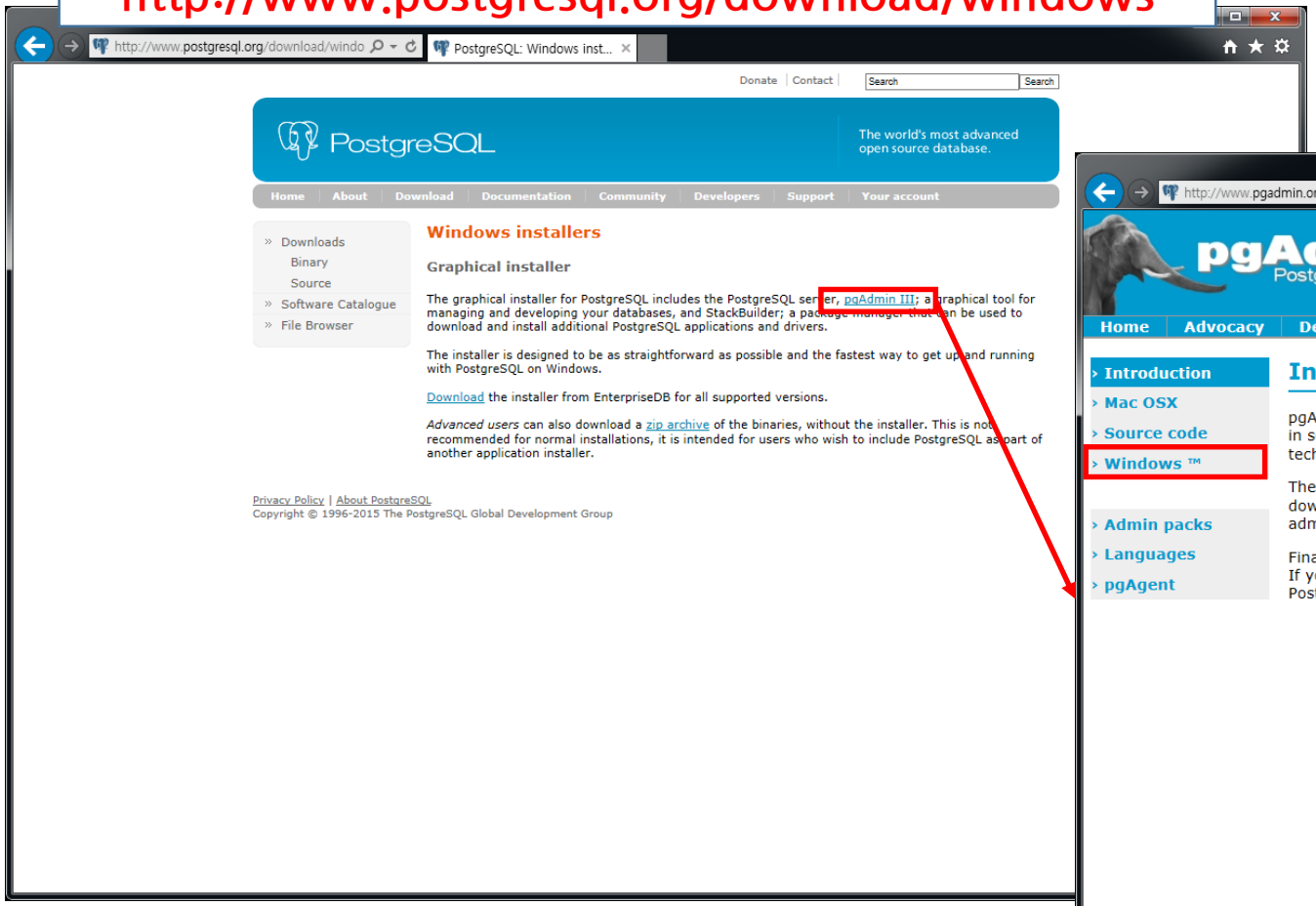


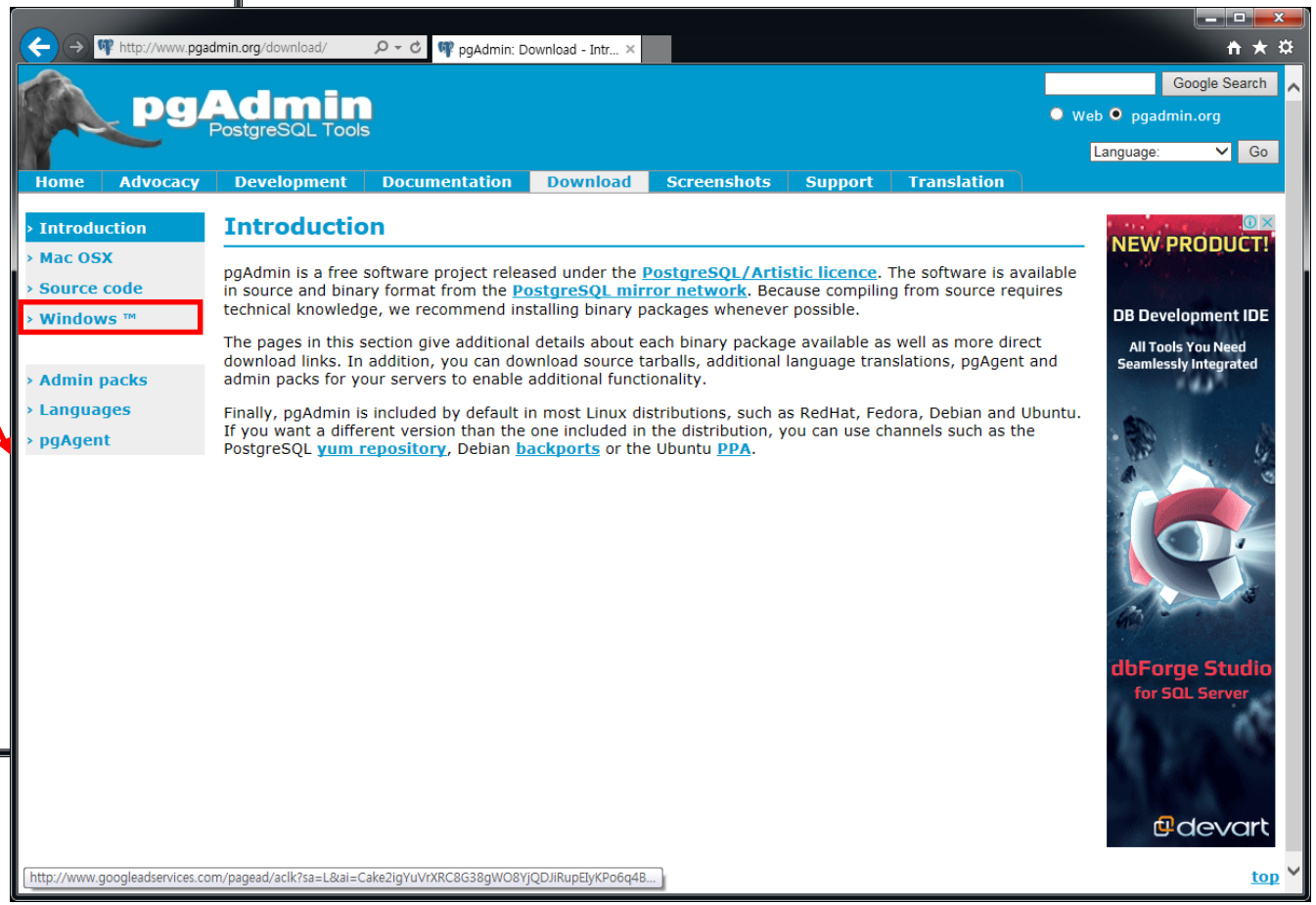
PostgreSQL Setup - 6



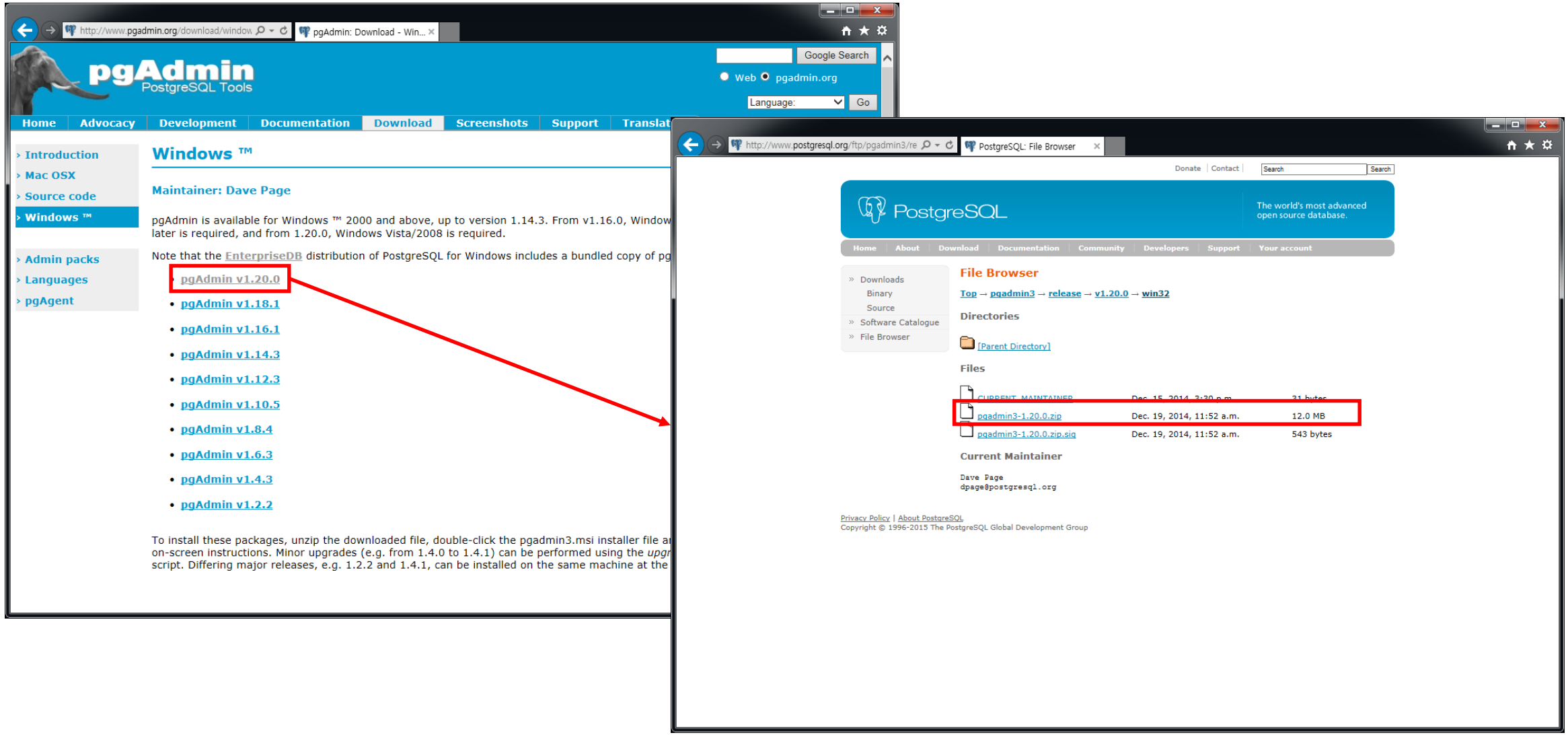
PG Admin Setup - 1

http://www.postgresql.org/download/windows

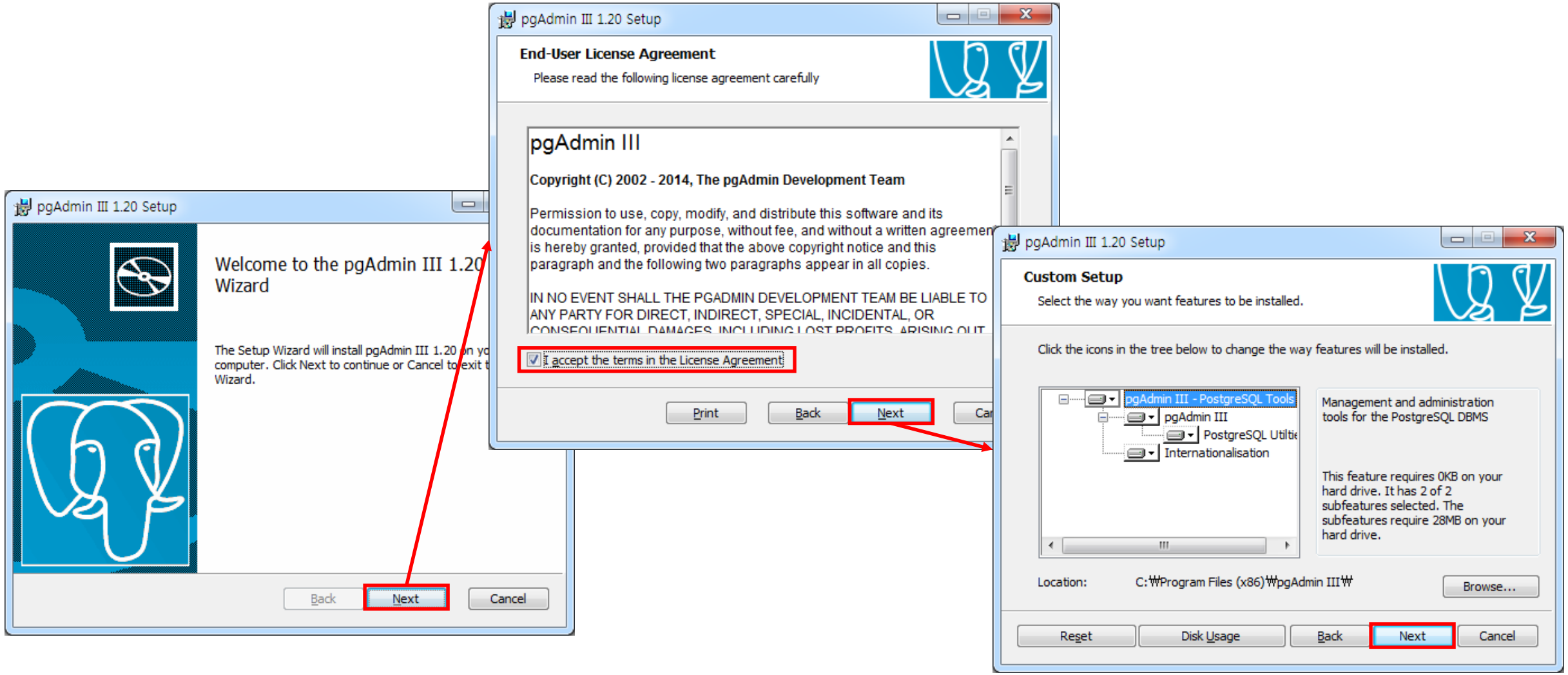




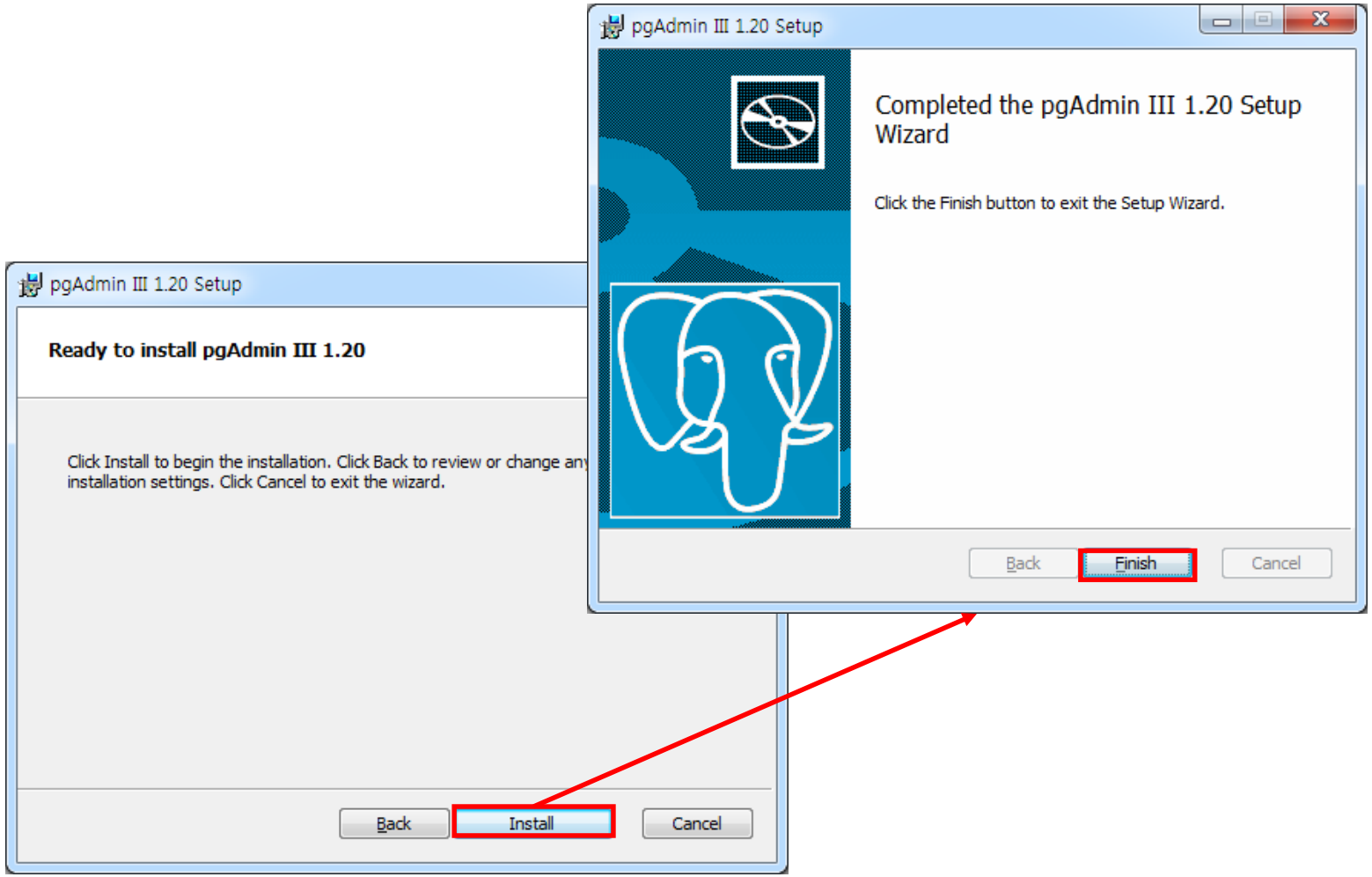
PG Admin Setup - 2



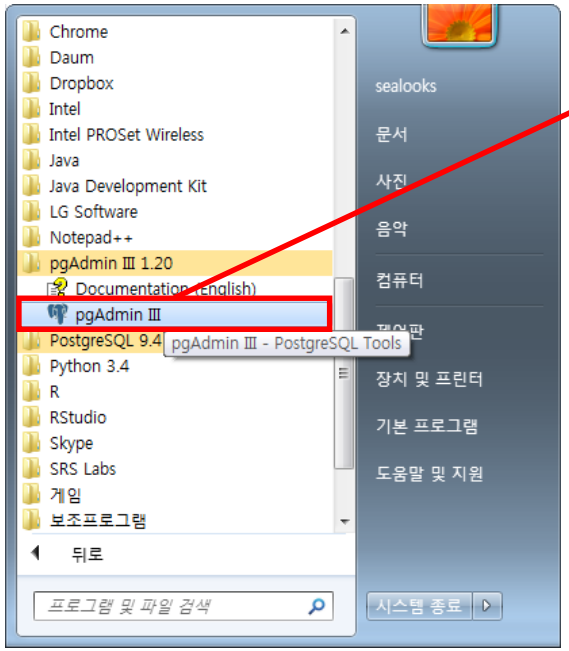
PG Admin Setup - 3



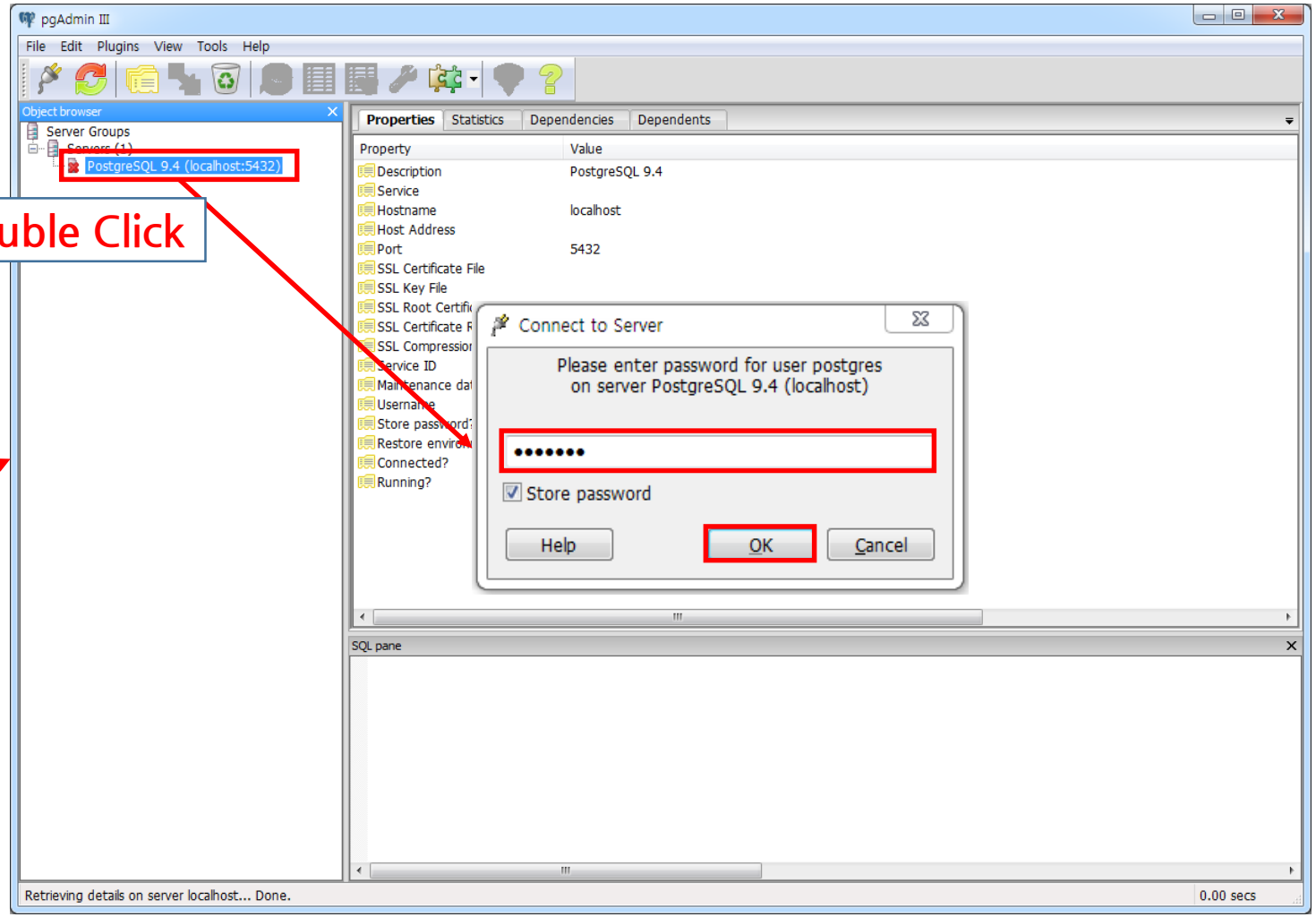
PG Admin Setup - 4



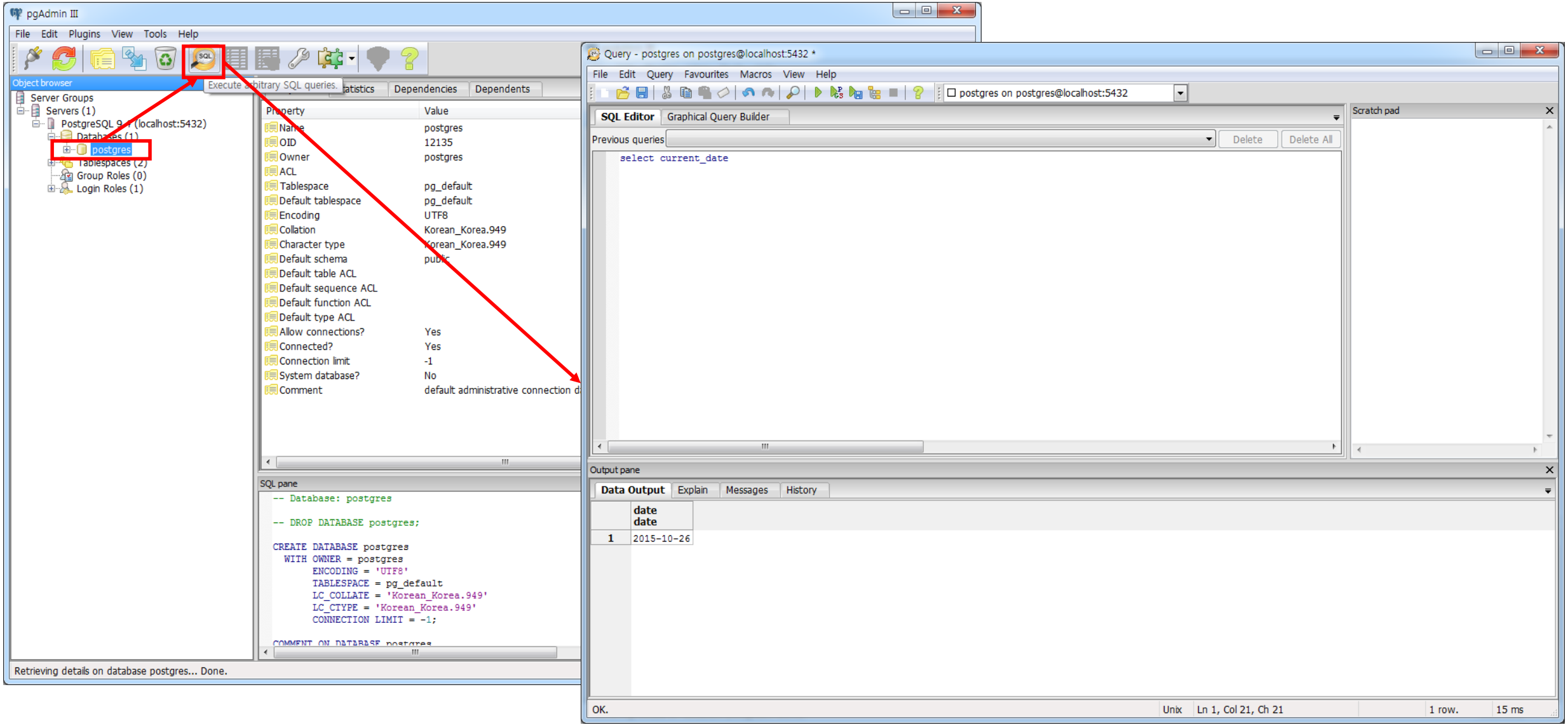
PG Admin Setup - 5



Double Click



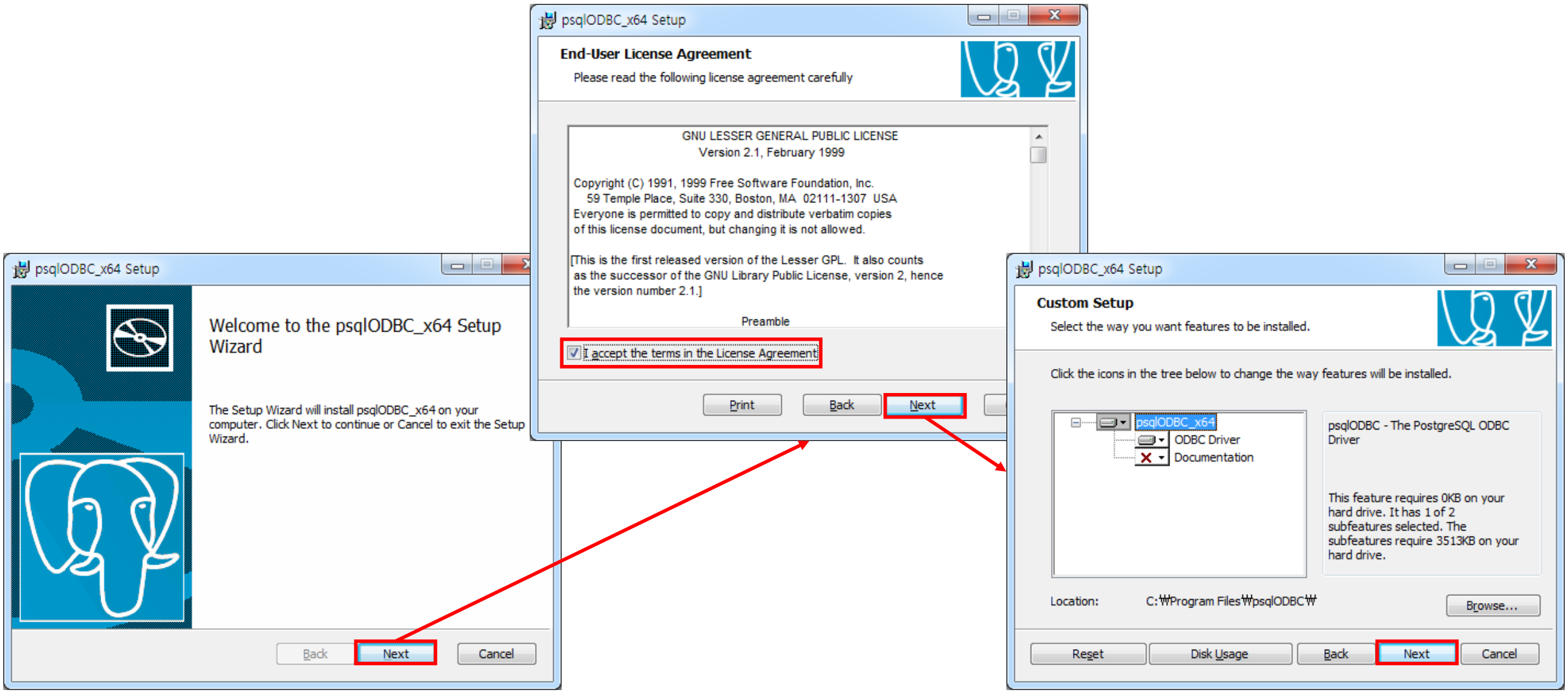
PG Admin Setup - 6



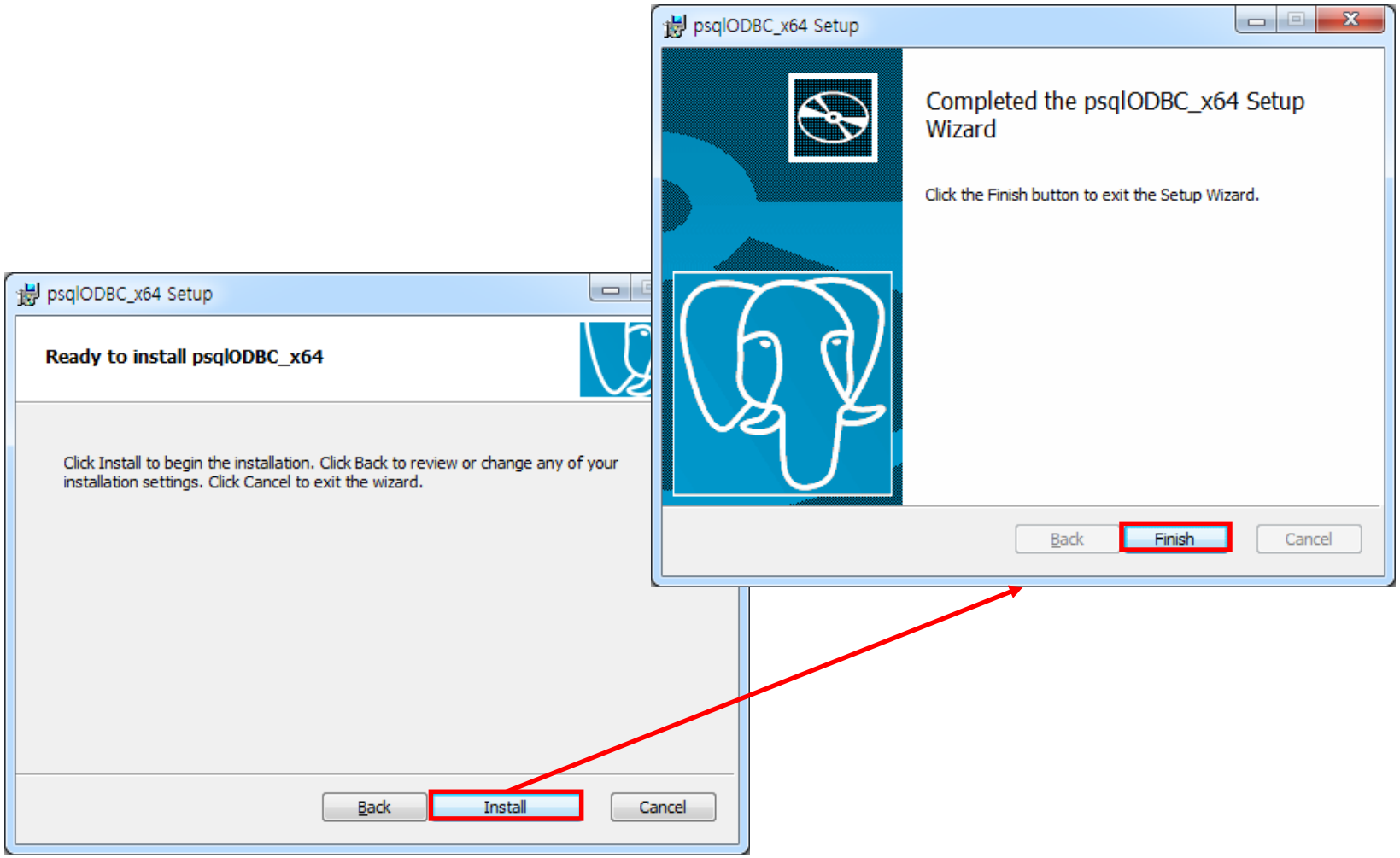
PostgreSQL ODBC Driver Setup - 1

http://www.postgresql.org/ftp/odbc/versions/msi/

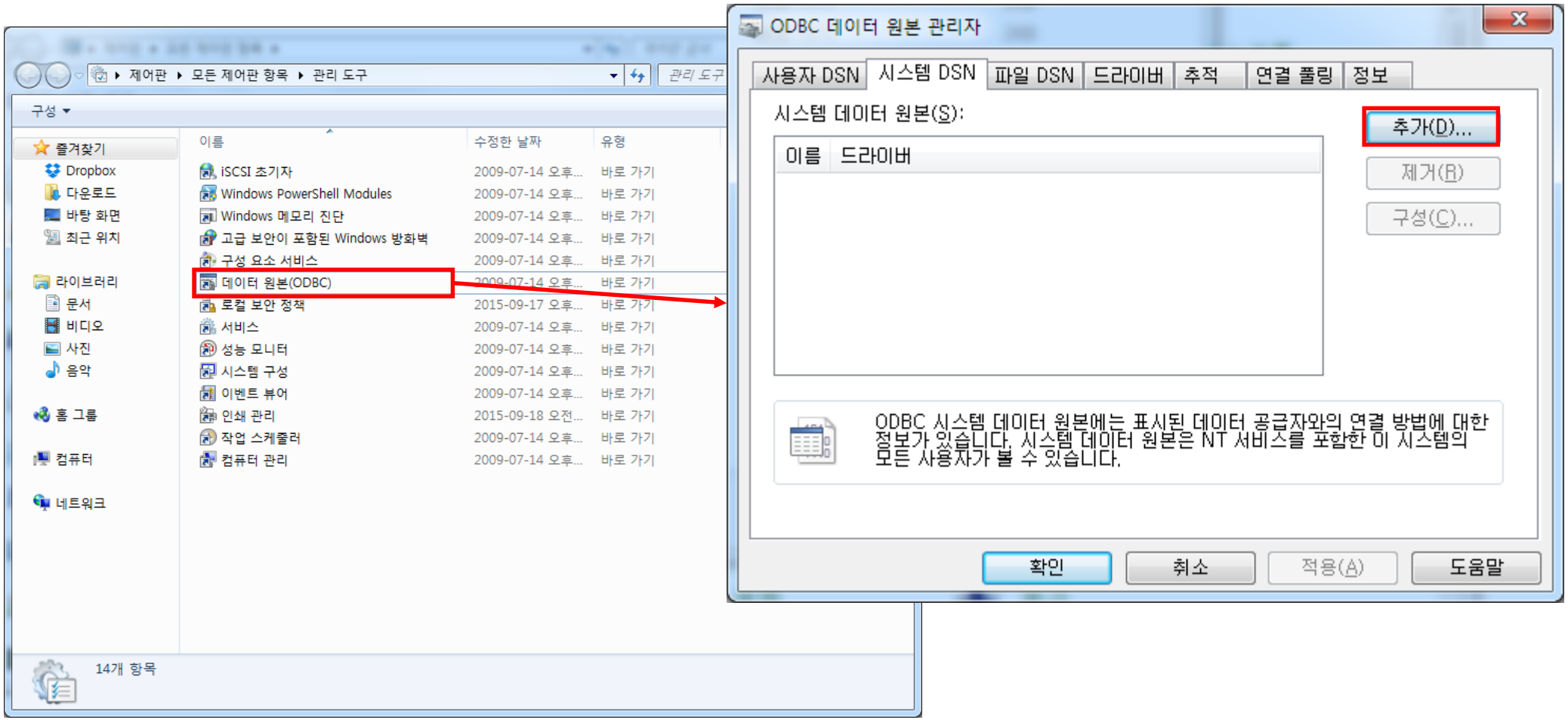
PostgreSQL ODBC Driver Setup - 2



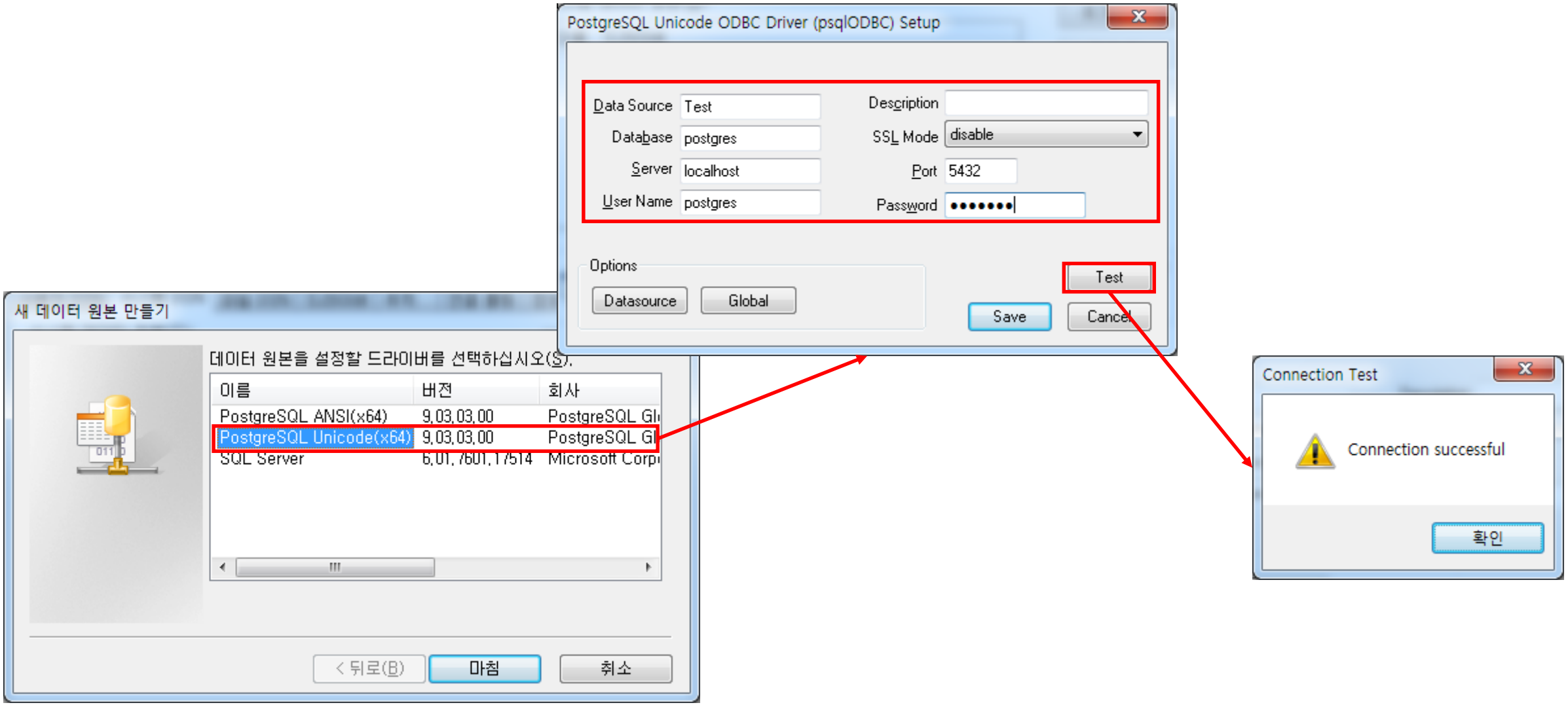
PostgreSQL ODBC Driver Setup - 3



PostgreSQL ODBC Driver Setup - 4



PostgreSQL ODBC Driver Setup - 5



SQL

왜 SQL인가?

- 2차원의 정형데이터를 다루는 가장 효율적인 언어.
  (Data를 사용하는 거의 모든 프로그램에서 SQL을 사용 하지 않고 효율적인 개발 불가)[DBMS, Hive, Spark, R(sqldf), …]
- Data의 처리 과정을 기술하는 방법이 아닌 원하는 Data 의 요건을 기술하는 방식.[SQL = Data 요건정의서]
  (단순성, 높은 생산성)
- Data의 요건이 기술된 요건 정의서 형태이므로 요건의 변경 시 적은 수정으로 요건을 반영할 수 있음.
  (효율적인 유지보수)

SQL 구문의 분류

- DDL(Data Definition Language : 데이터 정의 언어)
  - CREATE
  - DROP
  - ALTER
- DML(Data Manipulation Language : 데이터 조작 언어)
  - INSERT
  - UPDATE
  - DELETE
  - SELECT
- DCL(Data Control Language : 데이터 제어 언어)
  - GRANT
  - REVOKE
- …

기본 문법(SELECT)

- SELECT … 선택하고자 하는 속성목록을 기술.
- FROM … Source Data Set을 기술.
- [WHERE] … 선택하고자 하는 행의 조건의 목록을 기술.
- [GROUP BY … 집계하고자 하는 기준속성목록을 기술.
- [HAVING]] … 집계된 행을 기준으로 선택하고자 하는 행의 조건목록을 기술함.(GROUP BY가 선행조건)
- [ORDER BY] … 정렬의 기준이 되는 속성목록과 각 속성 목록의 정렬방식(올림차순, 내림차순)를 기술.
- [LIMIT] … 순서대로 반환되는 전체의 행 중 선택할 행의 수를 기술.

예시 데이터(테이블)

Table : Order

USRID	ORDER_DATE	METHOD	AMOUNT	DISCOUNT
INTEGER	VARCHAR(10)	VARCHAR(10)	INTEGER	NUMERIC
1	2015-08-01	CALL	10,000	-998.7
1	2015-08-03	TOUCH	10,000	
1	2015-08-10	TOUCH	10,000	-950.4
1	2015-08-14	CALL	10,000	-1,000.0
1	2015-08-25	TOUCH	10,000	
2	2015-08-03	TOUCH	5,000	-500.0
2	2015-08-11	TOUCH	5,000	-300.0
2	2015-08-12	TOUCH	5,000	-700.0
2	2015-08-22	TOUCH	5,000	-1,000.0
2	2015-08-28	TOUCH	5,000	-600.0
3	2015-08-07	CALL	10,000	-1,000.0
3	2015-08-19	TOUCH	10,000	-1,000.0
3	2015-08-30	CALL	10,000	-1,000.0
4	2015-08-05	CALL	20,000	-3,000.0
4	2015-08-18	TOUCH	30,000	-5,000.0
5	2015-08-15	CALL	10,000	-1,000.0
5	2015-08-17	CALL	10,000	
5	2015-08-21	CALL	10,000	-1,000.0
5	2015-08-23	CALL	10,000	-1,000.0
5	2015-08-29	CALL	10,000	-1,000.0

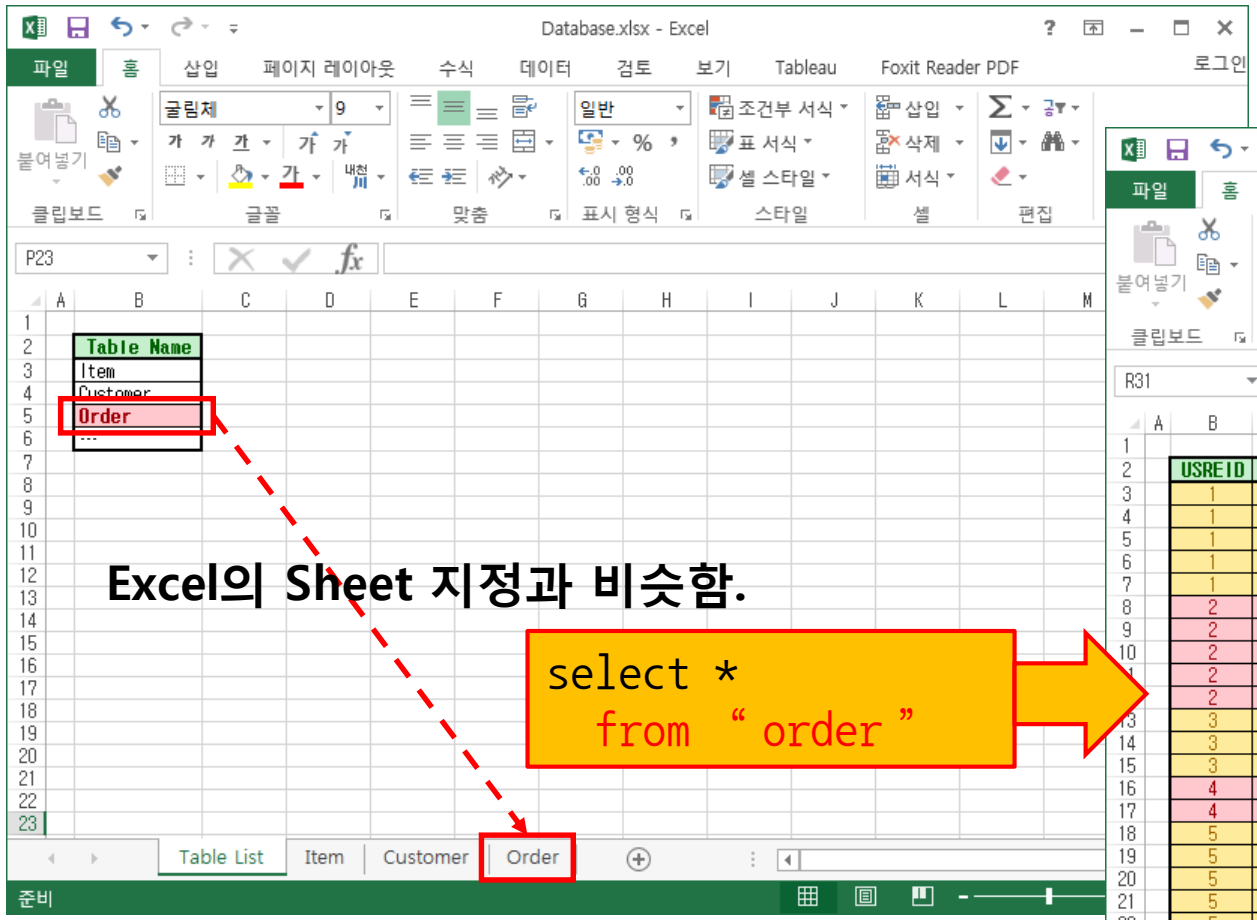
Data Script

예시 데이터(테이블)

```
create table "order" (
    userid     integer     not null,
    order_date varchar(10) not null,
    method     varchar(10) not null,
    amount     integer     not null,
    discount   numeric         null
);
```

```
insert into "order" values(1, '2015-08-01', 'CALL' , 10000, -998.7);
insert into "order" values(1, '2015-08-03', 'TOUCH', 10000, null);
insert into "order" values(1, '2015-08-10', 'TOUCH', 10000, -950.4);
insert into "order" values(1, '2015-08-14', 'CALL' , 10000, -1000);
insert into "order" values(1, '2015-08-25', 'TOUCH', 10000, null);
insert into "order" values(2, '2015-08-03', 'TOUCH',  5000, -500 );
insert into "order" values(2, '2015-08-11', 'TOUCH',  5000, -300 );
insert into "order" values(2, '2015-08-12', 'TOUCH',  5000, -700 );
insert into "order" values(2, '2015-08-22', 'TOUCH',  5000, -1000);
insert into "order" values(2, '2015-08-28', 'TOUCH',  5000, -600 );
insert into "order" values(3, '2015-08-07', 'CALL' , 10000, -1000);
insert into "order" values(3, '2015-08-19', 'TOUCH', 10000, -1000);
insert into "order" values(3, '2015-08-30', 'CALL' , 10000, -1000);
insert into "order" values(4, '2015-08-05', 'CALL' , 20000, -3000);
insert into "order" values(4, '2015-08-18', 'TOUCH', 30000, -5000);
insert into "order" values(5, '2015-08-15', 'CALL' , 10000, -1000);
insert into "order" values(5, '2015-08-17', 'CALL' , 10000, null);
insert into "order" values(5, '2015-08-21', 'CALL' , 10000, -1000);
insert into "order" values(5, '2015-08-23', 'CALL' , 10000, -1000);
insert into "order" values(5, '2015-08-29', 'CALL' , 10000, -1000);
```

FROM 절 – 출처는 어디야?



Excel의 Sheet 지정과 비슷함.

```
select *
  from "order"
```

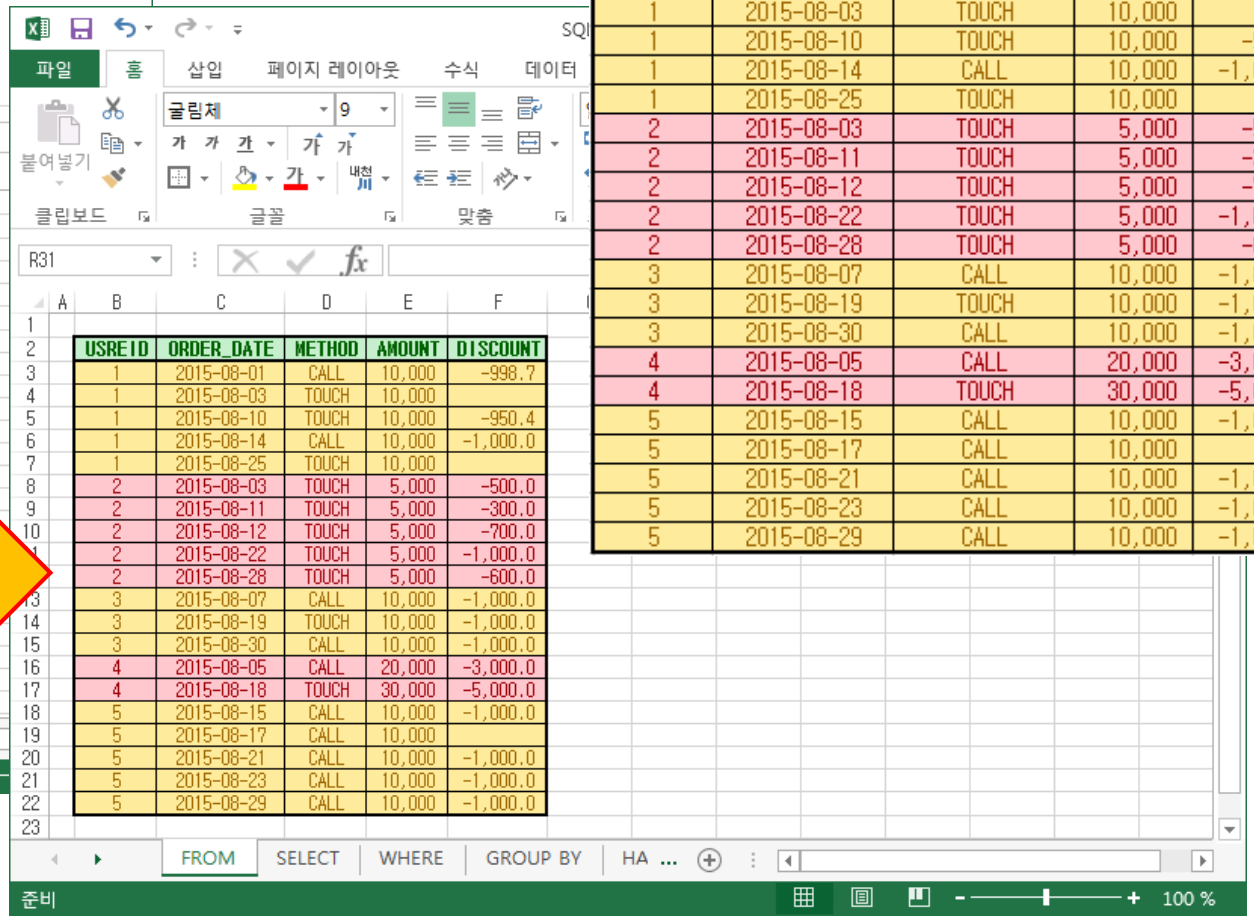


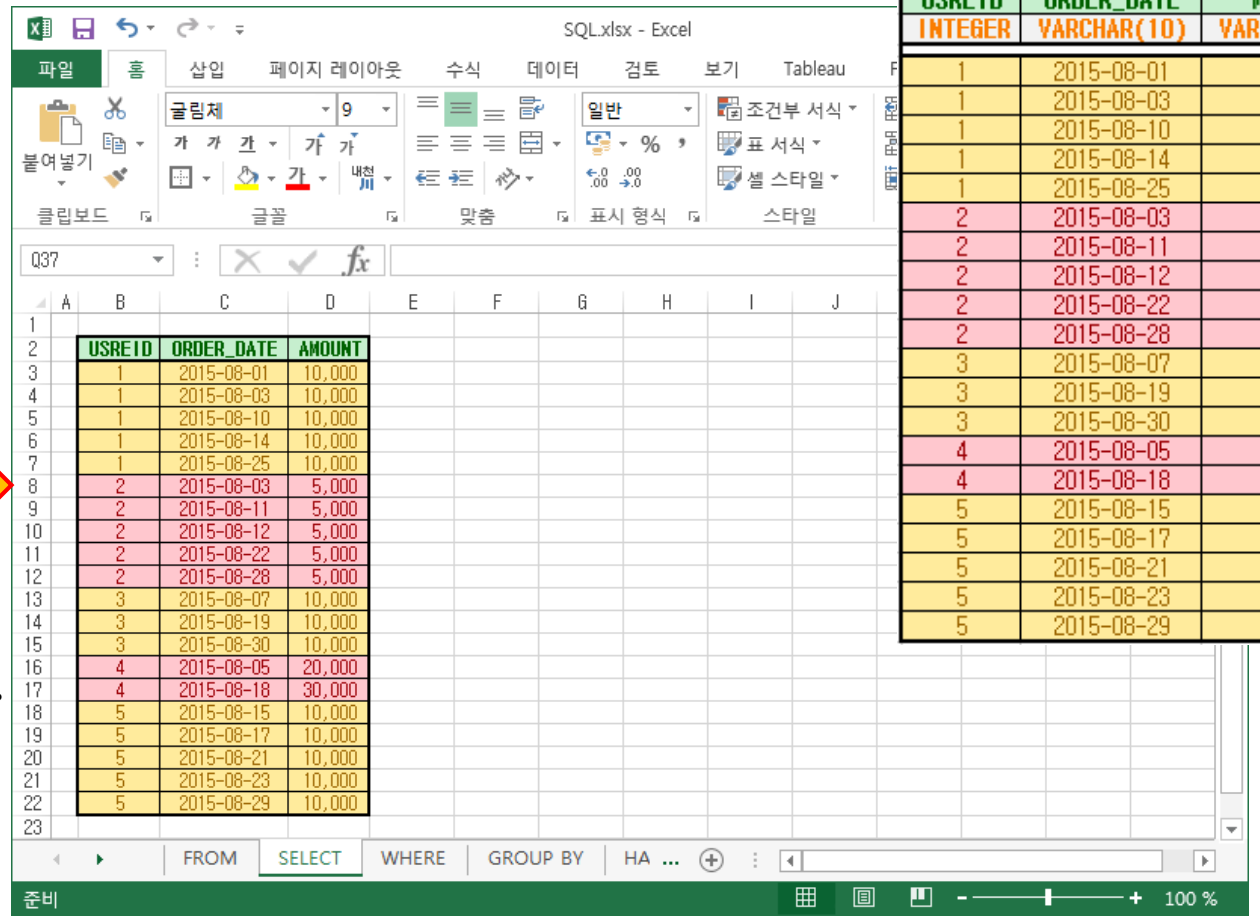
Table : Order

USREID	ORDER_DATE	METHOD	AMOUNT	DISCOUNT
INTEGER	VARCHAR(10)	VARCHAR(10)	INTEGER	NUMERIC
1	2015-08-01	CALL	10,000	-998.7
1	2015-08-03	TOUCH	10,000	
1	2015-08-10	TOUCH	10,000	-950.4
1	2015-08-14	CALL	10,000	-1,000.0
1	2015-08-25	TOUCH	10,000	
2	2015-08-03	TOUCH	5,000	-500.0
2	2015-08-11	TOUCH	5,000	-300.0
2	2015-08-12	TOUCH	5,000	-700.0
2	2015-08-22	TOUCH	5,000	-1,000.0
2	2015-08-28	TOUCH	5,000	-600.0
3	2015-08-07	CALL	10,000	-1,000.0
3	2015-08-19	TOUCH	10,000	-1,000.0
3	2015-08-30	CALL	10,000	-1,000.0
4	2015-08-05	CALL	20,000	-3,000.0
4	2015-08-18	TOUCH	30,000	-5,000.0
5	2015-08-15	CALL	10,000	-1,000.0
5	2015-08-17	CALL	10,000	
5	2015-08-21	CALL	10,000	-1,000.0
5	2015-08-23	CALL	10,000	-1,000.0
5	2015-08-29	CALL	10,000	-1,000.0

SELECT 절 – 어떤열을 볼까?

```
select userid
     , order_date
     , amount
  from “order”
```

Excel의 숨기기 기능과 비슷함.



USREID	ORDER_DATE	AMOUNT
1	2015-08-01	10,000
1	2015-08-03	10,000
1	2015-08-10	10,000
1	2015-08-14	10,000
1	2015-08-25	10,000
2	2015-08-03	5,000
2	2015-08-11	5,000
2	2015-08-12	5,000
2	2015-08-22	5,000
2	2015-08-28	5,000
3	2015-08-07	10,000
3	2015-08-19	10,000
3	2015-08-30	10,000
4	2015-08-05	20,000
4	2015-08-18	30,000
5	2015-08-15	10,000
5	2015-08-17	10,000
5	2015-08-21	10,000
5	2015-08-23	10,000
5	2015-08-29	10,000

Table : Order

USREID	ORDER_DATE	METHOD	AMOUNT	DISCOUNT
INTEGER	VARCHAR(10)	VARCHAR(10)	INTEGER	NUMERIC
1	2015-08-01	CALL	10,000	-998.7
1	2015-08-03	TOUCH	10,000	
1	2015-08-10	TOUCH	10,000	-950.4
1	2015-08-14	CALL	10,000	-1,000.0
1	2015-08-25	TOUCH	10,000	
2	2015-08-03	TOUCH	5,000	-500.0
2	2015-08-11	TOUCH	5,000	-300.0
2	2015-08-12	TOUCH	5,000	-700.0
2	2015-08-22	TOUCH	5,000	-1,000.0
2	2015-08-28	TOUCH	5,000	-600.0
3	2015-08-07	CALL	10,000	-1,000.0
3	2015-08-19	TOUCH	10,000	-1,000.0
3	2015-08-30	CALL	10,000	-1,000.0
4	2015-08-05	CALL	20,000	-3,000.0
4	2015-08-18	TOUCH	30,000	-5,000.0
5	2015-08-15	CALL	10,000	-1,000.0
5	2015-08-17	CALL	10,000	
5	2015-08-21	CALL	10,000	-1,000.0
5	2015-08-23	CALL	10,000	-1,000.0
5	2015-08-29	CALL	10,000	-1,000.0

WHERE 조건절 – 어떤 행을 볼까?

```
select *
  from "order"
 where userid = 2
```

Excel의 필터 기능과 비슷함.

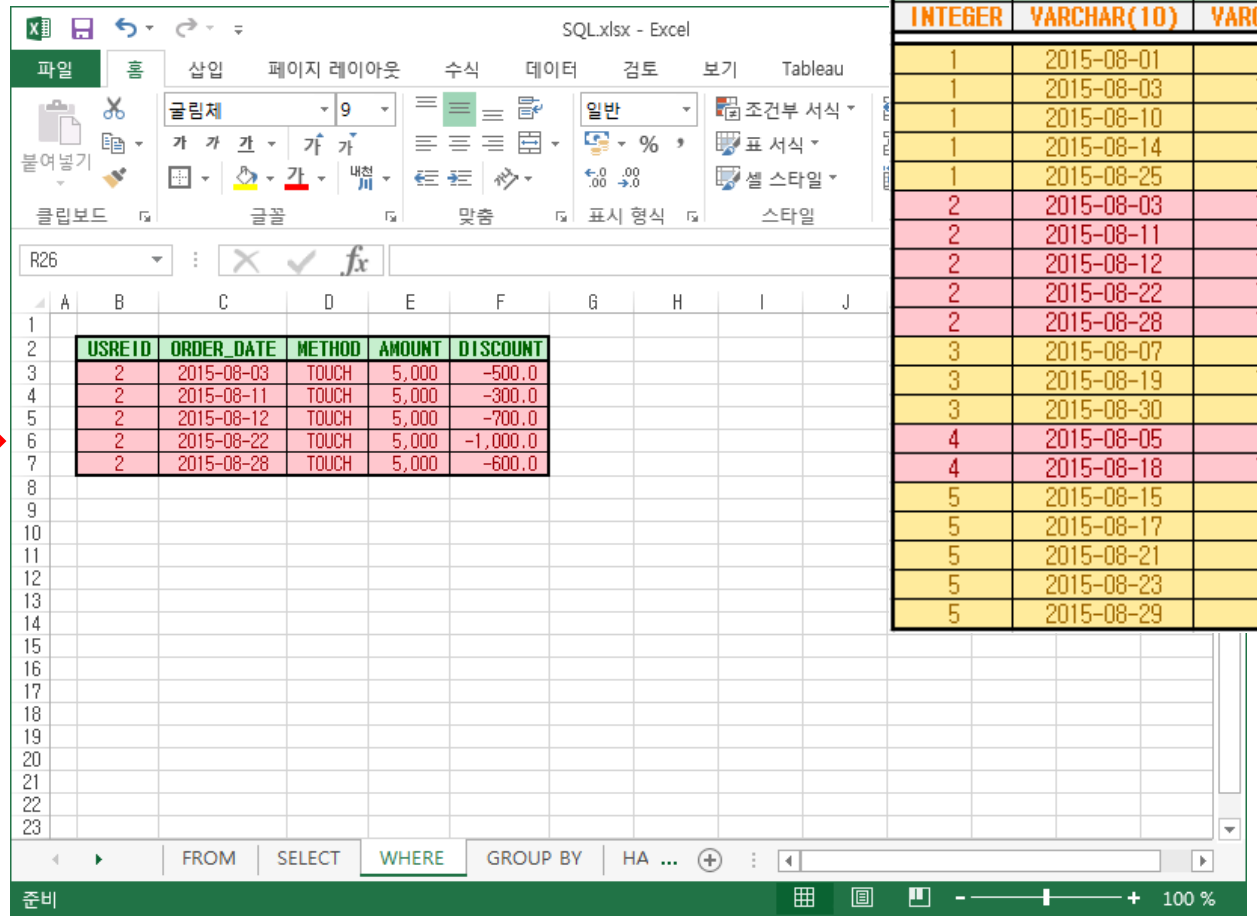


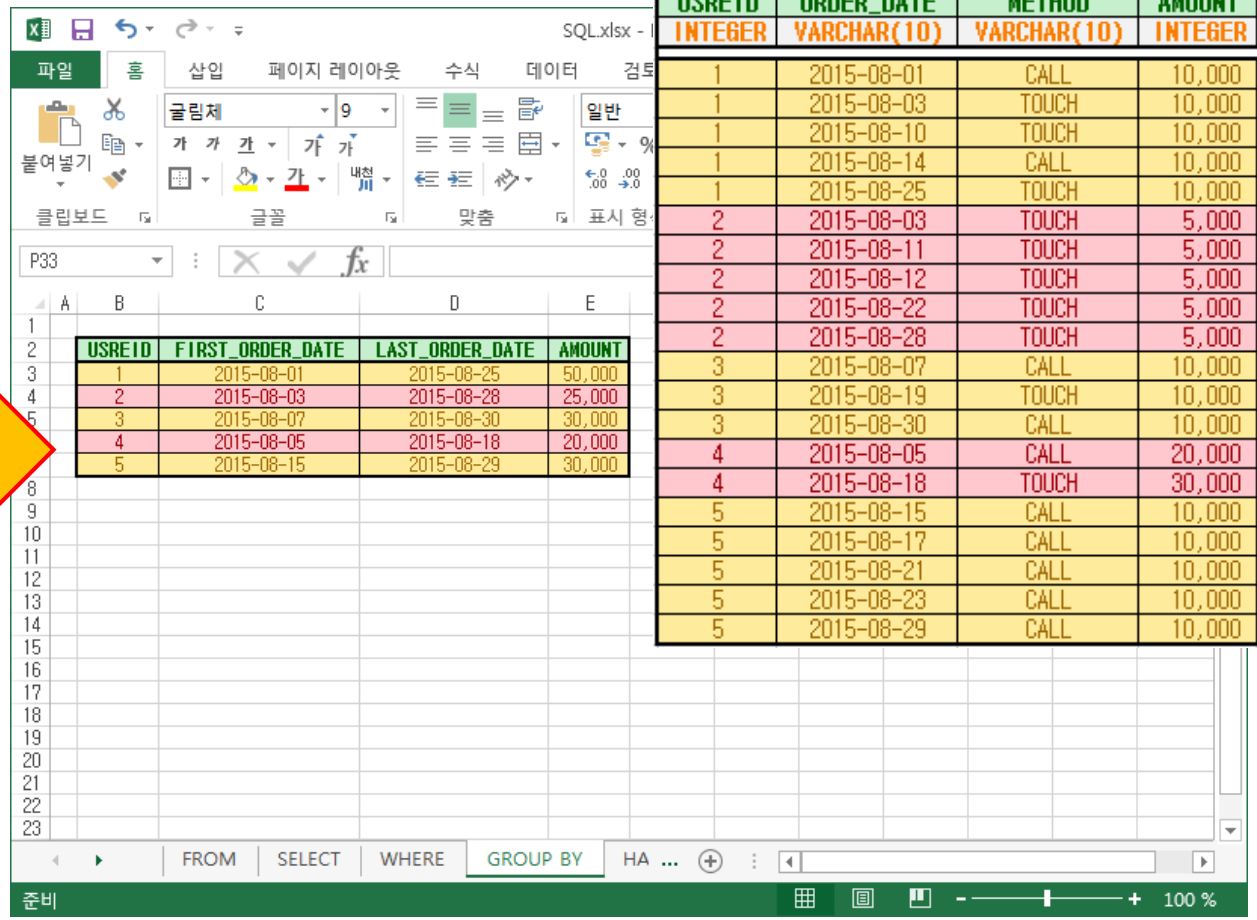
Table : Order

USRID	ORDER_DATE	METHOD	AMOUNT	DISCOUNT
INTEGER	VARCHAR(10)	VARCHAR(10)	INTEGER	NUMERIC
1	2015-08-01	CALL	10,000	-998.7
1	2015-08-03	TOUCH	10,000	
1	2015-08-10	TOUCH	10,000	-950.4
1	2015-08-14	CALL	10,000	-1,000.0
1	2015-08-25	TOUCH	10,000	
2	2015-08-03	TOUCH	5,000	-500.0
2	2015-08-11	TOUCH	5,000	-300.0
2	2015-08-12	TOUCH	5,000	-700.0
2	2015-08-22	TOUCH	5,000	-1,000.0
2	2015-08-28	TOUCH	5,000	-600.0
3	2015-08-07	CALL	10,000	-1,000.0
3	2015-08-19	TOUCH	10,000	-1,000.0
3	2015-08-30	CALL	10,000	-1,000.0
4	2015-08-05	CALL	20,000	-3,000.0
4	2015-08-18	TOUCH	30,000	-5,000.0
5	2015-08-15	CALL	10,000	-1,000.0
5	2015-08-17	CALL	10,000	
5	2015-08-21	CALL	10,000	-1,000.0
5	2015-08-23	CALL	10,000	-1,000.0
5	2015-08-29	CALL	10,000	-1,000.0

GROUP BY 절 – 어떤 기준으로 집계할까?

```
select userid
     , min(order_date) as first_order_date
     , max(order_date) as last_order_date
     , sum(amount)     as amount
  from "order"
 group by userid
```

Excel의 피벗 테이블 기능과 비슷함.



USREID	FIRST_ORDER_DATE	LAST_ORDER_DATE	AMOUNT
1	2015-08-01	2015-08-25	50,000
2	2015-08-03	2015-08-28	25,000
3	2015-08-07	2015-08-30	30,000
4	2015-08-05	2015-08-18	20,000
5	2015-08-15	2015-08-29	30,000

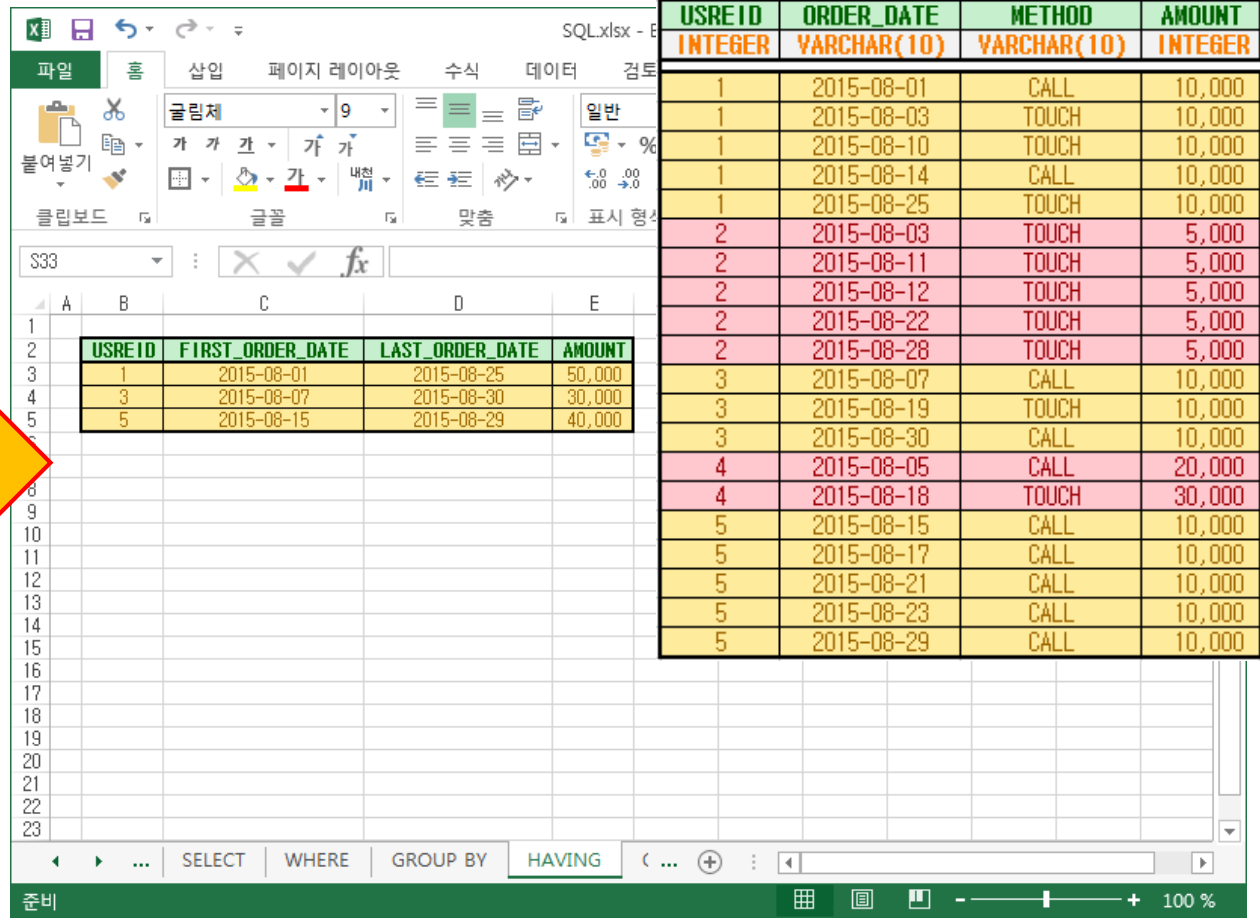
Table : Order

USREID	ORDER_DATE	METHOD	AMOUNT	DISCOUNT
INTEGER	VARCHAR(10)	VARCHAR(10)	INTEGER	NUMERIC
1	2015-08-01	CALL	10,000	-998.7
1	2015-08-03	TOUCH	10,000	
1	2015-08-10	TOUCH	10,000	-950.4
1	2015-08-14	CALL	10,000	-1,000.0
1	2015-08-25	TOUCH	10,000	
2	2015-08-03	TOUCH	5,000	-500.0
2	2015-08-11	TOUCH	5,000	-300.0
2	2015-08-12	TOUCH	5,000	-700.0
2	2015-08-22	TOUCH	5,000	-1,000.0
2	2015-08-28	TOUCH	5,000	-600.0
3	2015-08-07	CALL	10,000	-1,000.0
3	2015-08-19	TOUCH	10,000	-1,000.0
3	2015-08-30	CALL	10,000	-1,000.0
4	2015-08-05	CALL	20,000	-3,000.0
4	2015-08-18	TOUCH	30,000	-5,000.0
5	2015-08-15	CALL	10,000	-1,000.0
5	2015-08-17	CALL	10,000	
5	2015-08-21	CALL	10,000	-1,000.0
5	2015-08-23	CALL	10,000	-1,000.0
5	2015-08-29	CALL	10,000	-1,000.0

HAVING – 집계 후 어떤 행을 볼까?

```
select userid
     , min(order_date) as first_order_date
     , max(order_date) as last_order_date
     , sum(amount)     as amount
  from "order"
 group by userid
having sum(amount) >= 30000
```

Excel의 피벗 테이블 + 필터 기능과 비슷함.



USREID	FIRST_ORDER_DATE	LAST_ORDER_DATE	AMOUNT
1	2015-08-01	2015-08-25	50,000
3	2015-08-07	2015-08-30	30,000
5	2015-08-15	2015-08-29	40,000

Table : Order

USREID	ORDER_DATE	METHOD	AMOUNT	DISCOUNT
INTEGER	VARCHAR(10)	VARCHAR(10)	INTEGER	NUMERIC
1	2015-08-01	CALL	10,000	-998.7
1	2015-08-03	TOUCH	10,000	
1	2015-08-10	TOUCH	10,000	-950.4
1	2015-08-14	CALL	10,000	-1,000.0
1	2015-08-25	TOUCH	10,000	
2	2015-08-03	TOUCH	5,000	-500.0
2	2015-08-11	TOUCH	5,000	-300.0
2	2015-08-12	TOUCH	5,000	-700.0
2	2015-08-22	TOUCH	5,000	-1,000.0
2	2015-08-28	TOUCH	5,000	-600.0
3	2015-08-07	CALL	10,000	-1,000.0
3	2015-08-19	TOUCH	10,000	-1,000.0
3	2015-08-30	CALL	10,000	-1,000.0
4	2015-08-05	CALL	20,000	-3,000.0
4	2015-08-18	TOUCH	30,000	-5,000.0
5	2015-08-15	CALL	10,000	-1,000.0
5	2015-08-17	CALL	10,000	
5	2015-08-21	CALL	10,000	-1,000.0
5	2015-08-23	CALL	10,000	-1,000.0
5	2015-08-29	CALL	10,000	-1,000.0

ORDER BY – 순서대로 볼까?

```
select userid
     , min(order_date) as first_order_date
     , max(order_date) as last_order_date
     , sum(amount)     as amount
  from "order"
 group by userid
having sum(amount) >= 30000
 order by amount desc
```

Excel의 피벗 테이블 + 정렬 기능과 비슷함.

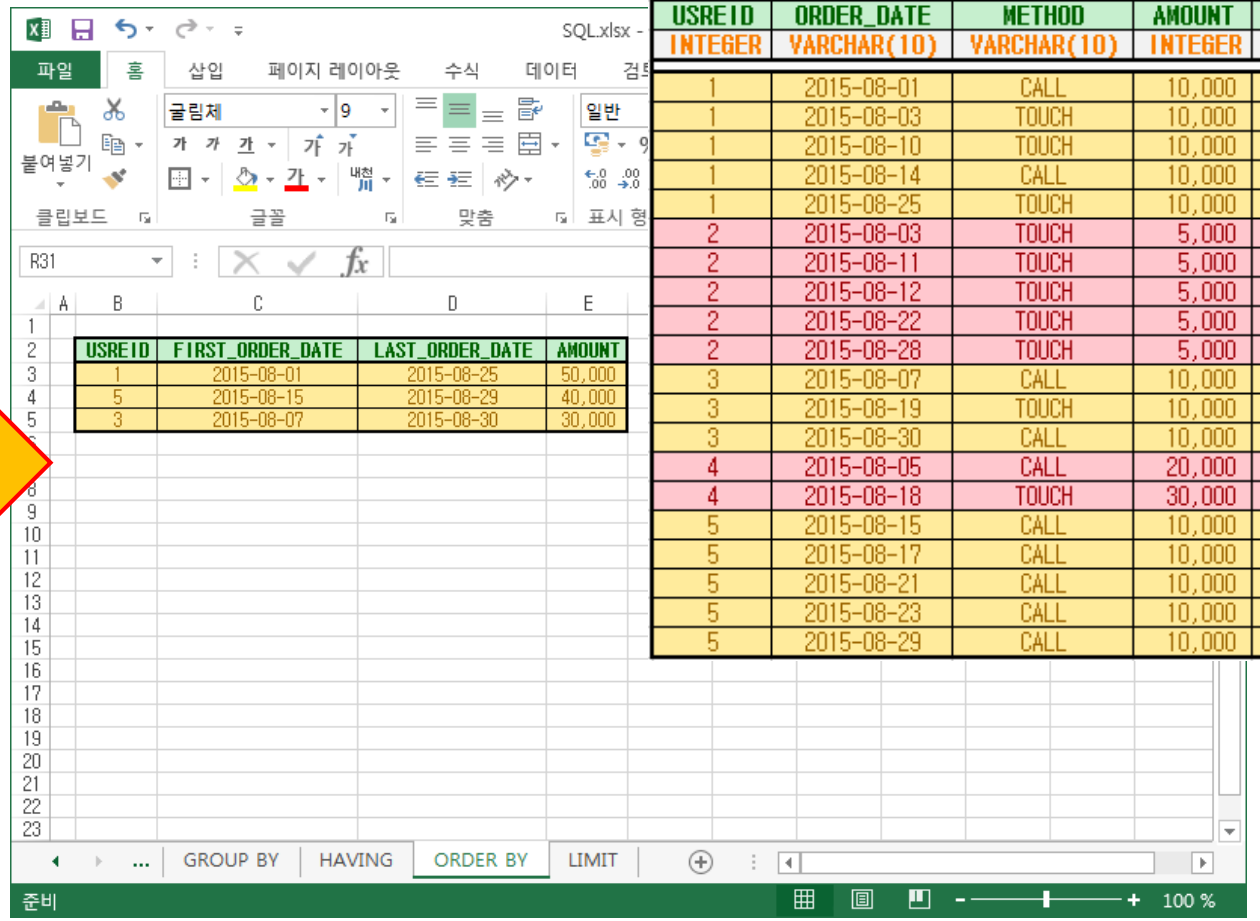


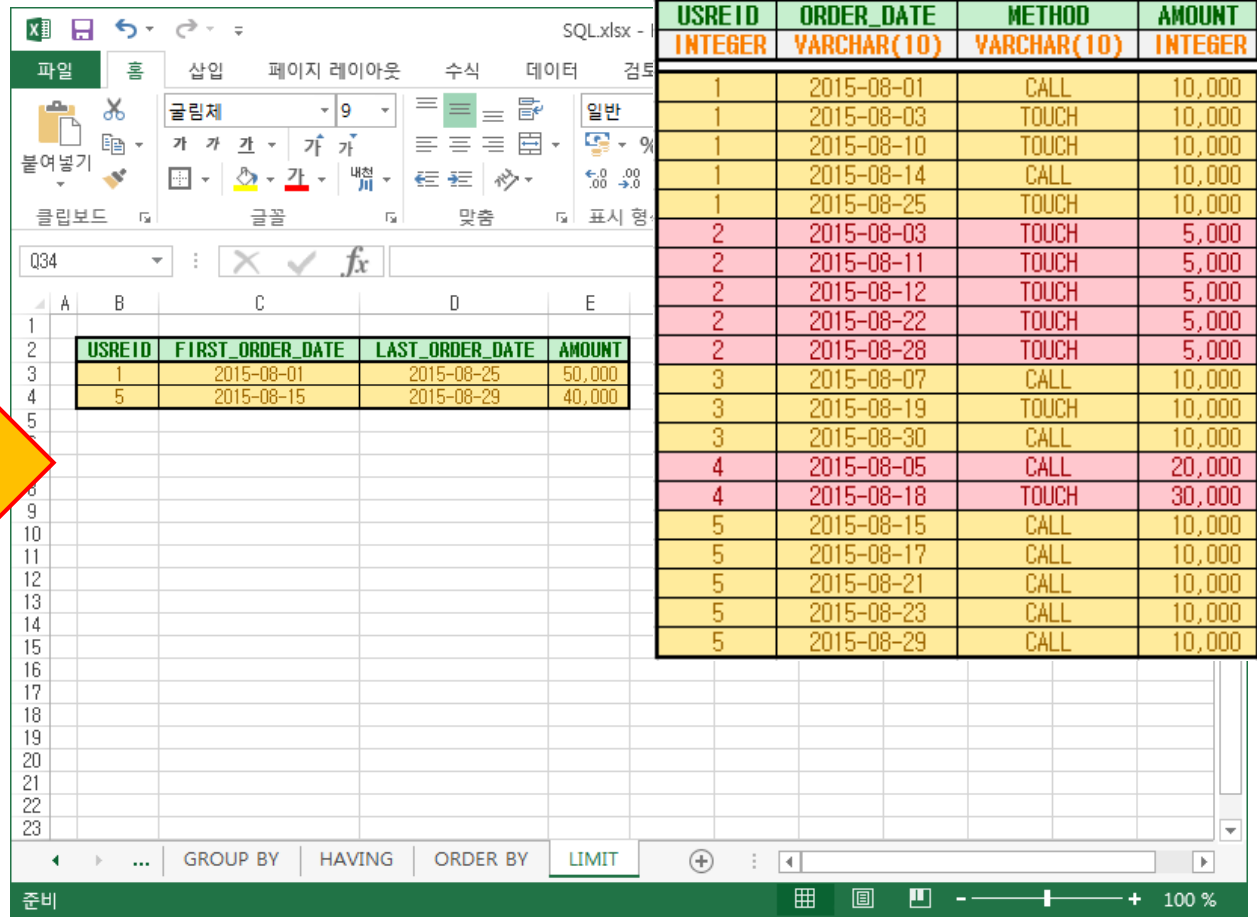
Table : Order

USREID	ORDER_DATE	METHOD	AMOUNT	DISCOUNT
INTEGER	VARCHAR(10)	VARCHAR(10)	INTEGER	NUMERIC
1	2015-08-01	CALL	10,000	-998.7
1	2015-08-03	TOUCH	10,000	
1	2015-08-10	TOUCH	10,000	-950.4
1	2015-08-14	CALL	10,000	-1,000.0
1	2015-08-25	TOUCH	10,000	
2	2015-08-03	TOUCH	5,000	-500.0
2	2015-08-11	TOUCH	5,000	-300.0
2	2015-08-12	TOUCH	5,000	-700.0
2	2015-08-22	TOUCH	5,000	-1,000.0
2	2015-08-28	TOUCH	5,000	-600.0
3	2015-08-07	CALL	10,000	-1,000.0
3	2015-08-19	TOUCH	10,000	-1,000.0
3	2015-08-30	CALL	10,000	-1,000.0
4	2015-08-05	CALL	20,000	-3,000.0
4	2015-08-18	TOUCH	30,000	-5,000.0
5	2015-08-15	CALL	10,000	-1,000.0
5	2015-08-17	CALL	10,000	
5	2015-08-21	CALL	10,000	-1,000.0
5	2015-08-23	CALL	10,000	-1,000.0
5	2015-08-29	CALL	10,000	-1,000.0

LIMIT – 몇 개 데이터만 볼까?

```
select userid
     , min(order_date) as first_order_date
     , max(order_date) as last_order_date
     , sum(amount)     as amount
  from "order"
 group by userid
Having sum(amount) >= 30000
 order by amount desc
 limit 2
```

Excel의 피벗 테이블 + 정렬 + 필터 기능과 비슷함.



USREID	FIRST_ORDER_DATE	LAST_ORDER_DATE	AMOUNT
1	2015-08-01	2015-08-25	50,000
5	2015-08-15	2015-08-29	40,000

Table : Order

USREID	ORDER_DATE	METHOD	AMOUNT	DISCOUNT
INTEGER	VARCHAR(10)	VARCHAR(10)	INTEGER	NUMERIC
1	2015-08-01	CALL	10,000	-998.7
1	2015-08-03	TOUCH	10,000	
1	2015-08-10	TOUCH	10,000	-950.4
1	2015-08-14	CALL	10,000	-1,000.0
1	2015-08-25	TOUCH	10,000	
2	2015-08-03	TOUCH	5,000	-500.0
2	2015-08-11	TOUCH	5,000	-300.0
2	2015-08-12	TOUCH	5,000	-700.0
2	2015-08-22	TOUCH	5,000	-1,000.0
2	2015-08-28	TOUCH	5,000	-600.0
3	2015-08-07	CALL	10,000	-1,000.0
3	2015-08-19	TOUCH	10,000	-1,000.0
3	2015-08-30	CALL	10,000	-1,000.0
4	2015-08-05	CALL	20,000	-3,000.0
4	2015-08-18	TOUCH	30,000	-5,000.0
5	2015-08-15	CALL	10,000	-1,000.0
5	2015-08-17	CALL	10,000	
5	2015-08-21	CALL	10,000	-1,000.0
5	2015-08-23	CALL	10,000	-1,000.0
5	2015-08-29	CALL	10,000	-1,000.0