Missi

on Impossible

Oracle SQL Developer

미션 임파서블 : Oracle SQL Developer 편

Oracle SQL Developer는 GUI 기반 워크벤치의 위력을 SQL과 PL/SQL로 확장한 것이다.
Oracle SQL Developer로 불가능해 보이는 작업을
완벽하게 처리해낸 과정을 소개한다.
Oracle SQL Developer의 유용성과 편의성, 파워를 직접 경험할 수 있을 것이다.

글 ‖ Jonathan Gennick (오라클 전문가) www.gennick.com

편리하고 유용한 작업대

훌륭한 장인은 좋은 작업대를 필요로 한다. 좋은 작업대(workbench)라면, 무릇 작업 공간에 여유가 있어야 할 것이다. 필요할 때 대로 바로 연장을 집을 수 있는가 하면, 필요 없을 때는 옆으로 제쳐놓을 수 있어야 한다. 한마디로, 좋은 작업대란 그것을 사용하는 사람의 확장이어야 한다.

오라클은 바로 이런 개념의 작업대를 Oracle SQL Developer라는 이름으로(코드명 Project Raptor) 발표했다. 이것은 깔끔하면서도 뛰어난 GUI 인터페이스를 갖춘 워크벤치이다. Oracle SQL Developer를 사용하면, 데이터베이스 스키마에서 모든 객체들을 브라우즈하고, SQL 문을 편집, 테스트, 포맷할 수 있으며, 데이터베이스 데이터를 편집하고, PL/SQL 블록을 디버그하고, 관리 리포트를 작성, 실행할 수 있다. 이외에도 더 많은 작업들이 가능하다.

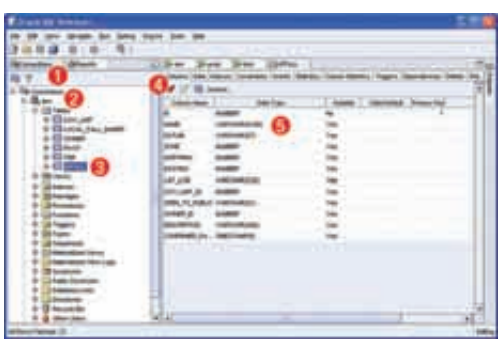
이 글에서는 Oracle SQL Developer를 사용해 여러 가지 일상 작업들을 신속하게 처리하는 방법에 대해 소개한다.

설치와 실행

Oracle SQL Developer를 설치하려면, oracle.com/technology/software/products/sql에서 각자 플랫폼에 적합한 버전을 다운로드한 후 압축을 풀면 된다.

Oracle SQL Developer를 실행하려면, 이 툴이 설치되어 있는 폴더를 열고 Oracle SQL Developer 실행파일(Microsoft Windows에서는 sqldeveloper.exe)을 더블클릭한다. Oracle SQL Developer가 두 개의 구획으로 나뉘어진 윈도로 열릴 것이다〈화면 1〉. 왼쪽 구획은 Connections 네비게이터를 보여준다. Connections 아이콘을 더블클릭하면, 새로운 커넥션이 만들어지는데, 이 새로운 커넥션의 이름과 사용자명을 입력하고, 타겟 호스트와 데이터베이스 서비스명을 채운 후, Connect 버튼을 클릭한다. 커넥션이 성공적으로 이루어지면, 오른편에 SQL Worksheet가 나타나게 된다.

〈화면 1〉 Oracle SQL Developer의 커넥션 네비게이터와 칼럼 탭



시스템에 오라클 클라이언트 소프트웨어와 tnsnames.ora 파일이 이미 설치되어 있다면, Oracle SQL Developer가 tnsnames.ora에 정의되어 있는 네트 서비스 명칭들로 Connections 네비게이터를 자동으로 채울 것이다. 이런 경우에도, Connections 네비게이터를 확장할 수 있는데, 타겟 데이터베이스로 연결하고자 원하는 네트 서비스명을 더블클릭하면 된다. (TNS_ADMIN 환경 변수를 사용해 tnsnames.ora 디렉토리로 지정해 주어야 하는 경우도 있다.)

이 기사를 위해 만든 o36sql.zip에는 setup.sql 스크립트가 포함되어 있다. 이 스크립트를 사용해 간단한 데이터를 만들어 보자. (o36sql.zip은 이 기사의 영문 웹사이트(www.oracle.com/technology/oramag/oracle/06-may/o36sql.html)에서 구할 수 있다.)

1. 테이블과 저장 프로시저를 만들 수 있는 스키마를 새로 만들거나 그런 스키마에 액세스할 수 있음을 확인한다.
2. Debug Any Procedure와 Debug Connect Session 시스템 권한을 스스로 부여하거나 DBA로부터 얻어낸다.
3. dev, test, prod라는 이름의 3가지 커넥션을 만든다. 그리고 1단계에서 만든 스키마에 이 커넥션들을 지정한다. (이 기사에서는 이 커넥션들을 별개 데이터베이스로 간주한다.)
4. dev 커넥션에서 오른쪽 버튼을 눌러 Open SQL Worksheet를 선택한다. 그러면, 이 워크시트가 오른편에 열릴 것이다.
5. 텍스트 에디터를 사용해 setup.sql 스크립트를 연다. 스크립트의 내용을 SQL 워크시트의 Enter SQL Statement 텍스트 상자에 붙인다.
6. F5 키를 눌러 스크립트를 실행한다.
7. 샘플 데이터가 만들어질 것이다. Script Output 탭에 피드백이 나타날 것이다.

시급한 과제의 신속한 해결

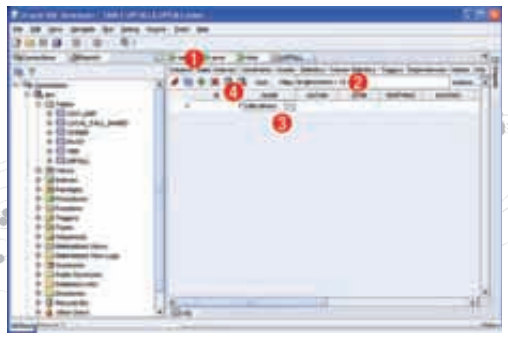
그러면, 이제 미국 미시간주 어퍼 페닌슐러(Upper Peninsula)의 폭포들에 대한 데이터베이스를 관리하고 있는 대학 연구소에서 일한다고 가정해 보자. 이 대학교는 오라클 데이터베이스를 사용하고 있는데, 당신은 이 연구소의 DBA이다. 정오에 업무상 중요한 약속이 있는데, 지금은 오전 11시 45분이다. 그런데, 과거의 경험상 정오까지 결코 끝낼 수 없을 것 같은 일들을 처리하도록 요구 받았다. 마침 어젯밤에 Oracle SQL Developer를 설치한 것이 생각났는데, 이것의 인터페이스와 기능들을 익히느라 몇 분만 투자하면, 아마도 정오 전에 일을 끝낼 수 있을 것 같은 좋은 예감이 들기 시작했다.

사무실 문을 닫은 뒤, 책상에 앉아서 Oracle SQL Developer를 시동한다. 첫 번째 작업은 개발중인 데이터베이스의 데이터베이스 칼럼의 크기를 점검하는 것이다. 개발자 중 한 명이 실수로 15 문자 길이 대신 150 문자 길이로 만들었다는 것이다.

신속한 작업을 위해 Oracle SQL Developer의 Connetions 네비게이터(〈화면 1〉의 1)의 Connections 노드를 펼친다. 커넥션 리스트를 살펴 dev 데이터베이스(〈화면 1〉의 2)의 커넥션 노드를 찾아 이를 펼친다. Tables 노드를 펼쳐 드릴다운을 계속해 UPFALL 테이블(〈화면 1〉의 3)을 클릭한다. 오른편에서 Columns 탭(〈화면 1〉의 4)을 클릭하자, 과연 Name 칼럼의 데이터 유형이 VARCHAR2(150)(〈화면 1〉의 5)로 나타난다. 칼럼 데이터 유형이 150 문자 길이라는 것이다.

칼럼 길이를 변경하기 전에, 데이터를 잠깐 점검해 보기로 한다. Data 탭(〈화면 2〉의 1)을 클릭하자, 모든 폭포의 리스트가 나타난다. 이 리스트를 보자, Name 필드의 크기를 줄였을 때 문제는 없을지 조금 걱정이 된다. 그래서 Filter 필드(〈화면 2〉의 2)에 'length(name) 〉15'를 입력하고 Enter를 누른다. 그러자, 우려한 대로, Little Miners Falls가 이름이 너무 긴 것으로 확인된다. 문자를 세어보니, 공백 포함해 적어도 4글자는 더 필요하다.

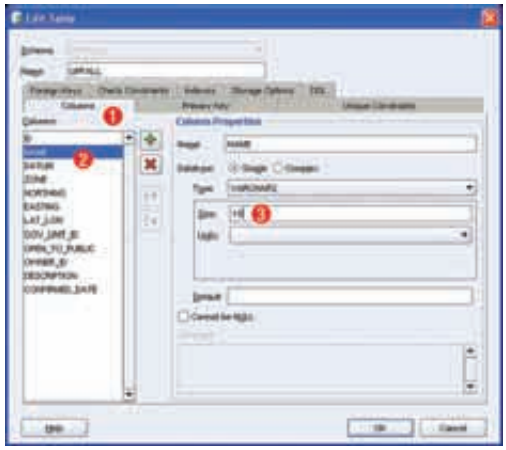
〈화면 2〉 Connections 네비게이터와 데이터 탭



지금 개발자들이 시험용으로 삽입한 데이터를 처리하고 있으므로, 자유롭게 변경할 수 있다. NAME 필드를 클릭하면 커서가 나타난다. 'Falls' 단어를 지우고 이름을 'Little Miners'로 남겨 둔다(〈화면 2〉의 3). (이렇게 단어를 지우는 것이 단순히 글자가 잘린 채 'Little Miners F'로 두는 것보다 낫다.) 그런 다음 Commit Changes 툴바 아이콘을 클릭해(〈화면 2〉의 4) 이 변경사항을 저장한다.

데이터의 모양을 좋게 유지하려면, Connections 네비게이터에서 UPFALL 테이블을 오른쪽 클릭한 후 Edit를 선택해 Edit Table 대화상자를 가져온다〈화면 3〉. 여기에서 Columns 탭(〈화면 3〉의 1)과 NAME 칼럼(〈화면 3〉의 2)을 클릭해 Size 필드의 값을 15로 변경한 후(〈화면 3〉의 3) OK를 클릭한다.

〈화면 3〉 Edit Table 대화상자



이로써 폭포 명칭 칼럼의 길이 문제는 해결되었다.

한 가지 업무가 끝난 셈이다.

현재 시각 11시 47분.

엑스포팅과 스크립팅

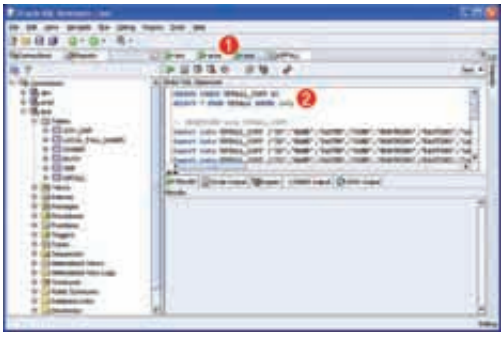
테스트 담당 기술자 한 명이 개발중인 폭포 테이블의 카피를 테스트 데이터베이스에 넣어달라고 요구했으므로, 이제 UPFALL 테이블의 Data 탭(〈화면 2〉의 1)을 살펴봐야 한다. 모든 로우를 복사하려 하므로, Filter 필드를 지우고 Enter를 클릭해 필터를 없앤다. 다음에, Connections 네비게이터의 UPFALL 테이블을 오른쪽 클릭한 후, Export -〉 SQL Insert를 선택해 엑스포트 프로세스를 시작한다. 다른 엑스포트 옵션으로는 CSV(Comma -Separated Values)와 XML, SQL Loader가 있다. 옮겨야 할 데이터가 많지 않으므로, SQL Insert 옵션을 사용해 INSERT 문 파일을 작성한다.

Export Table Data 대화상자〈화면 4〉가 나타날 것인데, 여기서 선택한 포맷(INSERT)과 출력 형태(File)를 확인하고, 파일 위치를 선택한다. Browse를 누르면 작성하려는 INSERT 파일에 적합한 위치를 네비게이트할 수 있다. 파일명으로 UPFALL_COPY.sql을 입력하고 Save를 클릭하면, Export Table Data 대화상자로 돌아가게 된다. 그 다음에 파일명 바로 위에 있는 Table 필드에 UPFALL_COPY를 입력한다. UPFALL_COPY는 이제 작성하려는 INSERT 문에서 테이블 명칭으로 사용될 것이다. 마지막으로, Apply를 클릭하면, UPFALL_COPY 테이블에 대한 일련의 INSERT로서 UPFALL 테이블의 데이터를 엑스포트할 것이다. 이제 모든 INSERT는 지정한 위치의 UPFALL_COPY.sql 파일에 기록될 것이다.

〈화면 4〉 Export Table 대화상자

이제 이전에 만들어 놓은 테스트 커넥션을 통해 액세스할 수 있는 테스트 데이터베이스에 데이터를 로드하자. 해당 테스트 커넥션의 오른쪽 버튼을 클릭해 Open SQL Worksheet를 선택하면, 테스트 커넥션을 위한 워크시트가 열리게 된다. 이제 액세스한 테이블과 커넥션에 대한 탭이 오른편 구획에 나타날 것이다(〈화면 5〉의 1).

〈화면 5〉 테스트 커넥션용 Connections 네비게이터와 SQL 워크시트



File -〉 Open File을 선택하면, UPFALL_COPY.sql 파일을 저장한 위치로 찾아갈 수 있는데, 여기서 파일명을 더블클릭하면 INSERT 문의 해당 스크립트를 편집 필드로 가져올 수 있다. 스크립트의 처음에 다음 문장을 추가해(〈화면 5〉의 2) UPFALL 테이블의 카피를 만든다.

```
CREATE TABLE UPFALL_COPY AS
SELECT * FROM UPFALL WHERE 1=2;
```

그 다음에 F5 키를 눌러 이 스크립트를 실행한다.

Script Output 탭의 컨텐츠가 훌륭해 보일 것이다. 테이블도 만들어졌고, INSERT도 모두 성공적이다. 테라바이트 용량의 데이터를 옮기는 데 이 방법을 사용하는 것이 별로 바람직하지는 않지만, 수십 로우를 옮기는 데는 아무런 문제가 없을 것이다. 점심 약속 시간에 맞출 가능성도

더 높다.

현재 시각 11시 50분.

질의와 최적화

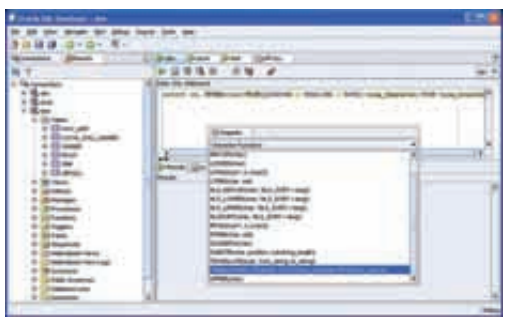
그 다음 작업은 유명한 폭포에 대한 리스트 작업 보고서를 만드는 개발자를 위해 질의문을 작성하는 것이다. Tools -〉SQL Worksheet를 선택해 dev에서 새로운 워크시트를 열고, 이것이 나타나면 dev 커넥션을 선택한다. 다음과 같이 입력해 질의를 시작해 보자.

```
select id,
```

그 개발자는 폭포 이름을 대문자로 쓰고, 앞이나 뒤에서 실수로 생긴 공백들을 지워주기를 바라고 있다. 맨오른쪽에 있는 Snippets 버튼을 클릭하면(또는 View -〉Snippet 선택), 코드 스니핏 윈도가 열리는데, 이 윈도는 데스크탑의 어디든 편한 곳으로 떼내거나 드랙해 화면 맨 위에 띄워놓을 수 있다〈화면 6〉. Snippets 윈도 상단의 리스트에서 Character 함수를 선택해, UPPER and TRIM 기능 프로토타입을 바로 질의문으로 드랙앤드롭한다. 이로써 간단한 편집 작업이 끝난다.

```
select id, UPPER(TRIM(BOTH ' '
FROM name)) name
from upfall;
```

〈화면 6〉 Connections 네비게이터, SQL 워크시트와 유동적인 Snippets 윈도



Ctrl-Enter를 누르면 이 문장을 실행할 수 있으며, Results 탭에서 출력 결과를 점검할 수 있다. 그 다음 한 줄 건너뛰어 두 번째 질의를 만드는데, 이번에는 LOCAL_FALL_NAMES 테이블에 대해서이다.

```
select id, UPPER(TRIM(BOTH ' ' FROM name)) name
from local_fall_names;
```

두 번째 질의에 커서가 있는 상태에서 Ctrl-Enter를 다시 한 번 누른다. 두 질의 모두 워크시트에서 볼 수 있지만, 커서가 있는 질의만 Ctrl-Enter에 반응해 수행된다. 이 두 번째 질의의 결과도 만족스러워 보인다.

그렇다면, 첫 번째 질의 끝에 있는 세미콜론을 지우고 질의들 사이에 있는 빈 줄에 UNION을 입력해, 두 개의 SELECT 문을 결합하는 하나의 질의를 만든다. Ctrl-Enter를 누르면, 수정된 질의를 수행할 수 있다. 결과를 보니 좋아 보인다.

이제 실행 계획(execution plan)을 잠깐 체크해 보자. Execute Explain Plan 툴바 아이콘을 클릭하면 실행 계획들이 나타날 것이다. 각 테이블도 모두 훑어볼 수 있는데, 이 테이블들은 질의를 수행했을 때 얻을 수 있는 결과이다. 이를 보니 실행 계획도 잘 된 것 같다.

이로써 개발자에게 질의문을 보낼 준비가 거의 다 되었지만, 그 전에 모양을 좀 다듬어보기로 한다. Ctrl-B를 누르면 Oracle SQL Developer가 급하게 작성한 질의문을 멋지게 포맷해 준다. 핵심어는 대문자로 바뀌고, 절은 들여쓰기 되며, 줄 간격도 일관되게 정리된다〈화면 7〉.

〈화면 7〉 재포맷된 SQL

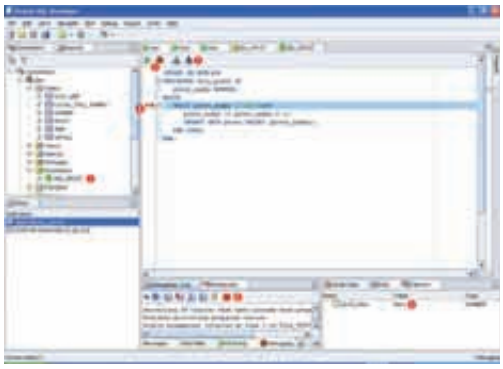
이제 다 잘 되었다. 질의문을 복사해 이메일 메시지에 붙여서 개발자에게 전송한다. 세 번째 작업이 끝난 것이다.

현재 시각 11시 53분.

디버깅

개발자가 부탁한 마지막 작업은 저장 프로시저의 디버깅을 도와 달라는 것이었다. Connections 네비게이터의 dev 커넥션으로 돌아가, Procedures 노드를 펼쳐서 FILL_PIVOT 프로시저(〈화면 8〉의 1)를 클릭한다. 그러면, 이 프로시저의 텍스트가 Code 탭에 나타날 것이다. 이 프로시저는 PIVOT 테이블을 1부터 1,000까지 1,000개의 로우로 채우도록 설정되어 있는데, 지금 당장은 어떤 로우도 만들지 않고 있는 것이 문제였다.

〈화면 8〉 FULL_PIVOT 프로시저의 디버깅



얼핏 보아서는 FILL_PIVOT의 코드는 문제 없어 보였고, 1,000개의 로우를 만들어낼 것처럼 보였다. 그래서 수행 과정을 직접 따라가 보기로 하고, 디버거를 사용해 실제로 어떻게 되는지 살펴보았다. CodeDebug 아이콘(〈화면 8〉의 2) 밑의 Edit 아이콘을 클릭한다. 다음에 Compile for Debug 툴바 아이콘(〈화면 8〉의 3)을 클릭하고, WHILE 라인(〈화면 8〉의 4) 왼편의 여백을 클릭해 브레이크포인트를 만든다.

그 다음에 Debug 툴바 아이콘(〈화면 8〉의 2)을 클릭하면, 대화상자가 열리면서 디버깅 작업이 시작된다. 매개변수 값이 필요한 경우에는 프롬프트에 값을 채워 넣어야 한다. 추가해야 할 매개변수가 없으므로 OK를 클릭한다.

이제 실행이 시작되고, 브레이크포인트에 도달하면 바로 멈춘다. Smart Data 탭에서 현재 변수 값을 점검해 볼 수 있다. 그 결과, pivot_index 변수가 비어 있음(〈화면 8〉의 5)을 발견한다. 이것이 바로 문제였다. Debugging - Log 탭에 있는 사각형의 빨간 Terminate 버튼(〈화면 8〉의 6)을 클릭해, pivot_index의 선언을 변수를 초기화하도록 바꾼다.

```
pivot_index NUMBER := 0
```

이렇게 변경한 뒤, Compile 버튼(〈화면 8〉의 7)을 클릭한다. 이번에는 Comile for Debug가 아니라는 점에 유의하자. 그 다음에는 녹색 화살표를 클릭해 프로시저를 실행한다. Connections 네비게이터의 PIVOT 테이블을 선택해 테이블의 Data 탭을 보면, 이제 1,000개의 로우가 제대로 만들어지고 있음을 확인할 수 있다.

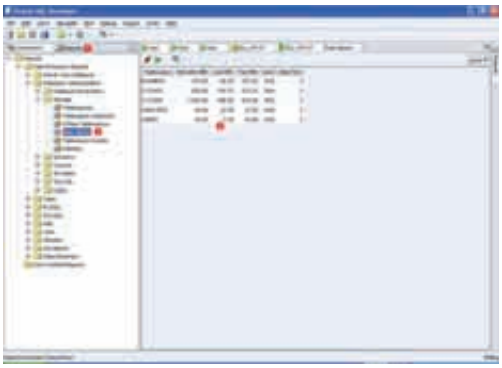
디버거는 다양한 종류의 코드 버그를 찾아내고 수정하는 작업을 간단하게 끝내는 데 상당한 도움이 된다. 미소를 머금고 업무 리스트에서 또 하나를 지운다.

현재 시각 11시 55분.

리포팅

이제 점심 약속을 향해 내닫기 전에 한 가지 일을 더 처리하려 한다. 늘 현재 실행중인 데이터베이스의 여유 공간에 대해 다소 걱정해온 것이 사실이다. 이를 점검하기 위해 왼편의 Reports 탭(〈화면 9〉의 1)을 클릭해 Reports 네비게이터를 펼친다. 이 리포트 리스트에서 Reports, Data Dictionary Reports, Database Administration 및 Storage 등을 거쳐 마침내 Free Space 리포트(〈화면 9〉의 2)에 이른다. 이것을 클릭하면, 커넥션을 선택하라고 할 것이다. 이번에는 prod 커넥션을 선택한다. 그러면, Enter Bind Values 대화상자가 열리면서, 현재 공란으로 놓아둔 테이블스페이스 명칭을 입력하라고 할 것인데, 왜냐하면 모든 테이블스페이스의 여유 공간을 보기 원하기 때문이다. Apply 버튼을 클릭하면 Free Space 리포트(〈화면 9〉의 3)가 나타난다. 지금 당장 시급한 문제는 없어 보인다. 모든 테이블스페이스가 조금 더 커져도 될 여유가 있는 것 같다.

〈화면 9〉 Free Space 리포트의 실행



마지막 작업을 마친 뒤 시간을 보자 안도의 한숨이 절로 나온다. 화창한 일요일이다. 선글라스를 집어 들고 약속 시간에 넉넉하게 집을 나선다.

현재 시각 11시 56분.

매우 생산적인 멀티태스킹 공간

앞에서 소개한 이야기는 물론 픽션이지만, 이것은 현실적으로 충분히 가능한 이야기이다. 데이터베이스 관리자와 개발자는 끊임없이 밀어닥치는 불시의 질의문 작성, 데이터베이스 테이블의 변경, 버그 추적, 애플리케이션 개발 등의 요구를 처리해야 하며, 일상적으로 데이터베이스를 관리해야 한다. 때때로 진행중이던 태스크는 '지금 당장' 수행되어야 하는, 우선순위가 높은 다른 태스크 때문에 중단되어야 한다. 이렇게 대기중인 태스크들이 쌓여갈수록 태스크의 유형은 더 빈번하게 변하게 되며, 통합되지 않은 툴세트로 이들을 관리하기는 더 어렵게 된다.

Oracle SQL Developer는 소프트웨어 작업대이다. 비단 한 번에 한 작업이 아니라 동시에 여러 작업을 펼쳐놓을 수 있는 공간을 제공한다. 강력한 툴들이 오른쪽 버튼 클릭이나 툴바 아이콘 형태로 작업대에 정리되어 있어, 필요할 때 바로 사용할 수 있고, 필요 없을 때는 옆으로 젖혀 놓으면 된다. 또, 하나의 태스크에서 다음 태스크로 넘어갈 때 그 흐름을 쉽게 관리할 수 있으므로, 생산성은 대폭 높아질 것이고 스트레스는 현격히 줄어들 것이다. 오라클 데이터베이스를 사용해본 경험이 있든 아니면 처음 사용하든 관계없이 Oracle SQL Developer는 SQL과 PL/SQL 코드를 개발, 디버그하고 데이터베이스를 관리하기에 매우 효율적인 작업대를 제공해 줄 것이다. 약속 시간을 지키는 데 도움이 될 것임은 물론!

이 글의 원문은 http://www.oracle.com/technology/oramag/oracle/06-may/o36sql.html을 참조하기 바랍니다.

Next Steps

- **Oracle SQL Developer :** www.oracle.com/technology/products/database/sql_developer/index.html
- **SQL Tools :** OKM 2006년 봄호 '업무 효율성 높여주는 SQL 툴들'
- **Oracle SQL Developer의 다운로드 :** www.oracle.com/technology/software/products/sql/index.html

참고도서

[SQL*Plus User's Guide and Reference]
[SQL*Plus Quick Reference]
[Oracle Database PL/SQL User's Guide and Reference]
[Oracle Database SQL Reference]