

Tugas Besar

aditya luthfi maulana harahap

Desember 2019

Chapter 1

Membuat Aplikasi Menggunakan Oracle Apex

1.1 Oracle Application Express (APEX)

Oracle Application Express (Oracle APEX) yang dulu disebut HTML-DB adalah sebuah framework yang berbasis pada sebuah database dedicated (sementara ini sampai versi terbaru masih dedicated untuk Oracle Db saja dan lisensi include dalam lisensi database), ini artinya apa bahwa engine aplikasi dibangun sepenuhnya didalam sebuah database. Bahkan untuk arsitektur Embedded PL/SQL Gateway seperti yang dipakai dalam Oracle XE dan Oracle 11G file image (library,css,theme,dll) disimpan didalam database metadata juga. Inilah hal yang berbeda dibandingkan framework yang lain.

1.1.1 Cara Mengakses APEX

Oracle APEX dapat di akses secara online ataupun offline.

1. Offline

Dapat di download disini : <https://www.oracle.com/tools/downloads/apex-downloads.html>

2. Online

Dapat di akses disini : <https://apex.oracle.com/en/>

Disini saya menggunakan APEX online. untuk mengakses nya silahkan klik link yang telah disediakan diatas. Setelah masuk ke situsnya anda haruslah memiliki akun APEX online terlebih dahulu untuk mengaksesnya.

Jika anda belum memiliki akun silahkan daftar dengan mengklik *Get Started for Free*, lalu isi data-data yang diminta. Jika telah memiliki akun silahkan login dengan klik *sign in* dan masukkan *Workspace*, *Username*, *Password* anda.

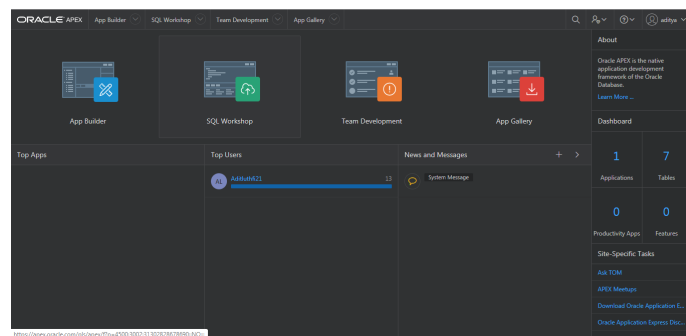
1.2 Merancang database di APEX Online

1.2.1 Membuat Database di APEX

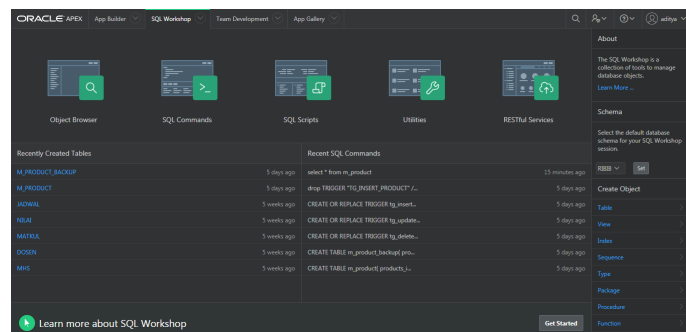
Untuk membuat database bisa upload tabel melalui excel dan juga dibuat langsung melalui *SQL Commads*, tetapi disini saya ditugaskan untuk membuat tabel melalui *SQL Commads*.

Berikut langkah-langkah untuk menjalankan *SQL Commands* :

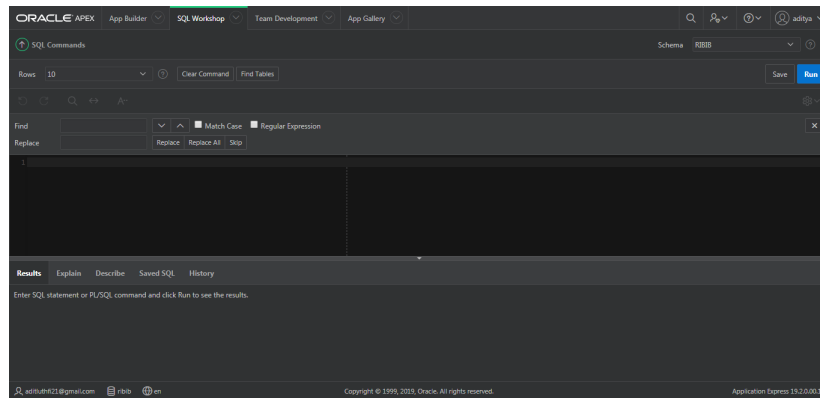
1. Klik SQL Workshop



2. Lalu Klik SQL Commands



3. Berikut adalah tampilan SQL Commands, dan di sini Anda memasukkan perintah-perintah SQL.



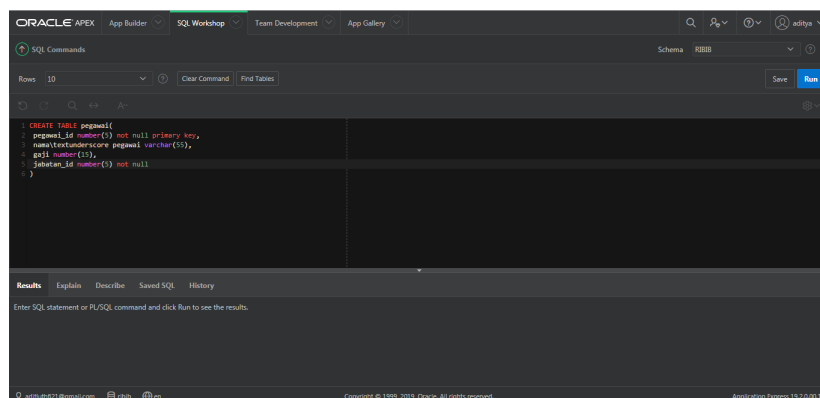
1.2.2 Membuat tabel database

Membuat table pegawai

Table ini berfungsi untuk menampung data pegawai

Ketikkan query berikut di SQL Command :

```
CREATE TABLE pegawai(  
  pegawai_id number(5) not null primary key,  
  nama_pegawai varchar(55),  
  gaji number(15),  
  jabatan_id number(5) not null  
)
```



hasilnya akan seperti ini :

PEGAWAI				
Table	Data	Indexes	Model	Constraints
Add Column Modify Column Rename Column Drop Column Rename Copy Drop Truncate Create Lookup Table Create App				
Column Name	Data Type	Nullable	Default	Primary Key
PEGAWAI_ID	NUMBER(5,0)	No	-	1
NAMA_PEGAWAI	VARCHAR2(55)	Yes	-	-
GAJI	NUMBER(15,0)	Yes	-	-
JABATAN_ID	NUMBER(5,0)	No	-	-

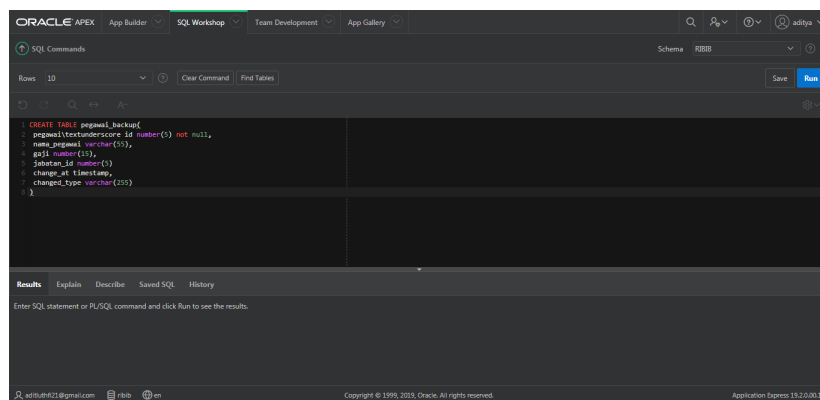
[Download](#) | [Print](#)

Membuat table pegawai_backup

Table ini berfungsi sebagai backup dari table pegawai apabila record dari table pegawai di manipulasi(insert,update,delete)

Ketikkan query berikut di SQL Command :

```
CREATE TABLE pegawai_backup(
pegawai_id number(5) not null,
nama_pegawai varchar(55),
gaji number(15),
jabatan_id number(5)
change_at timestamp,
changed_type varchar(255) )
```



hasilnya akan seperti ini :

PEGAWAI BACKUP

+ v

Table

Data

Indexes

Model

Constraints

Grants

Statistics

UI Defaults

Triggers

Dependencies

SQL

REST

Sample Queries

Add Column

Modify Column

Rename Column

Drop Column

Rename

Copy

Drop

Truncate

Create Lookup Table

Create App

Column Name	Data Type	Nullable	Default	Primary Key
PEGAWAI_ID	NUMBER(5,0)	No	-	-
NAMA_PEGAWAI	VARCHAR2(55)	Yes	-	-
GAJI	NUMBER(15,0)	Yes	-	-
JABATAN_ID	NUMBER(5,0)	Yes	-	-
CHANGED_AT	TIMESTAMP(6)	Yes	-	-
CHANGED_TYPE	VARCHAR2(255)	Yes	-	-

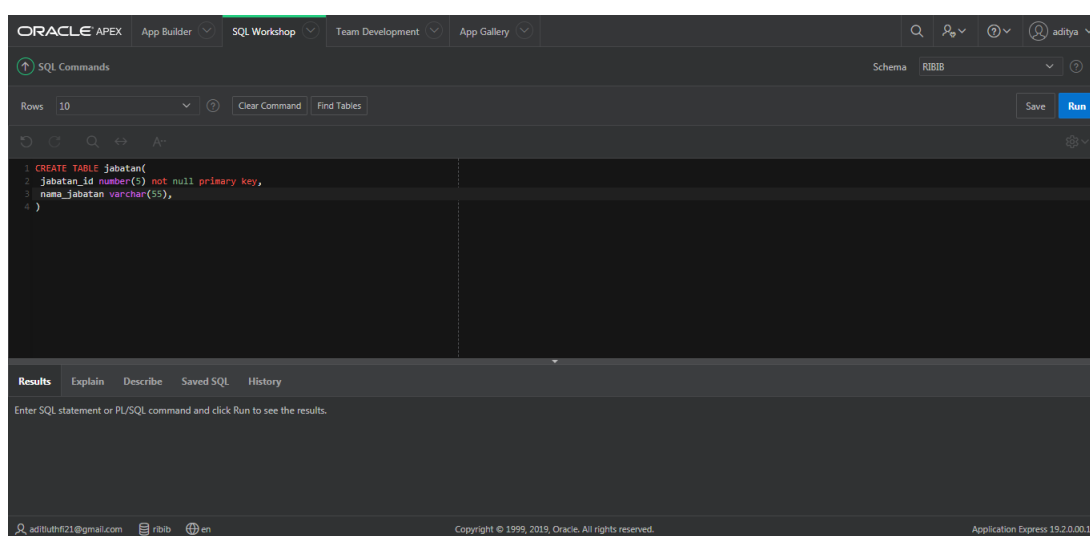
[Download](#) | [Print](#)

Membuat Table jabatan

Table ini berfungsi untuk menampung data jabatan.

Ketikkan query berikut di SQL Command :

```
CREATE TABLE jabatan(  
jabatan_id number(5) not null primary key,  
nama_jabatan varchar(55),  
)
```



hasilnya akan seperti ini :

JABATAN

+ v

Table

Data

Indexes

Model

Constraints

Grants

Statistics

UI Defaults

Triggers

Dependencies

SQL

REST

Sample Queries

Add Column

Modify Column

Rename Column

Drop Column

Rename

Copy

Drop

Truncate

Create Lookup Table

Create App

Column Name	Data Type	Nullable	Default	Primary Key
JABATAN_ID	NUMBER(5,0)	No	-	1
NAMA_JABATAN	VARCHAR2(55)	Yes	-	-

[Download](#) | [Print](#)

Membuat table jabatan_backup

Table ini berfungsi sebagai backup dari table jabatan apabila record dari table jabatan di manipulasi(insert,update,delete)

Ketikkan query berikut di SQL Command :

```
CREATE TABLE JABATAN_BACKUP
( JABATAN_ID NUMBER(5,0) NOT NULL ENABLE,
  NAMA_JABATAN VARCHAR2(55),
  CHANGED_AT TIMESTAMP,
  CHANGED_TYPE VARCHAR2(25)
)
```

```
1 CREATE TABLE JABATAN_BACKUP
2 ( JABATAN_ID NUMBER(5,0) NOT NULL ENABLE,
3   NAMA_JABATAN VARCHAR2(55),
4   CHANGED_AT TIMESTAMP,
5   CHANGED_TYPE VARCHAR2(25)
6 )
```

hasilnya akan seperti ini :

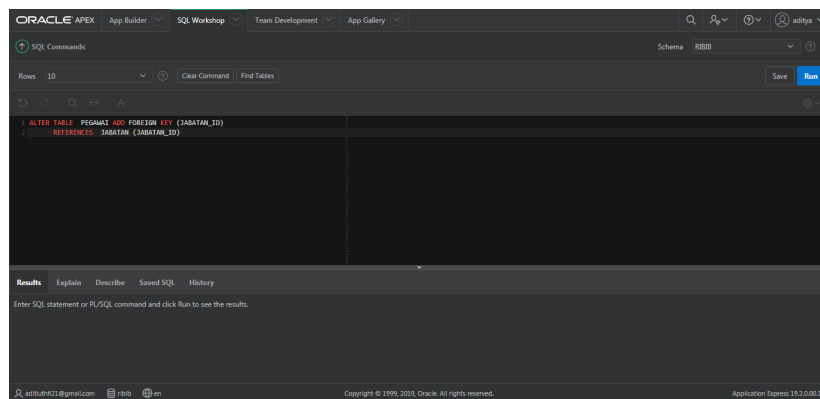
JABATAN_BACKUP													+ ▾
Table	Data	Indexes	Model	Constraints	Grants	Statistics	UI Defaults	Triggers	Dependencies	SQL	REST	Sample Queries	
Add Column	Modify Column	Rename Column	Drop Column	Rename	Copy	Drop	Truncate	Create Lookup Table	Create App				
Column Name			Data Type			Nullable		Default		Primary Key			
JABATAN_ID			NUMBER(5,0)			No		-		-			
NAMA_JABATAN			VARCHAR2(55)			Yes		-		-			
CHANGED_AT			TIMESTAMP(6)			Yes		-		-			
CHANGED_TYPE			VARCHAR2(25)			Yes		-		-			
Download Print													

1.2.3 Membuat relasi antar table

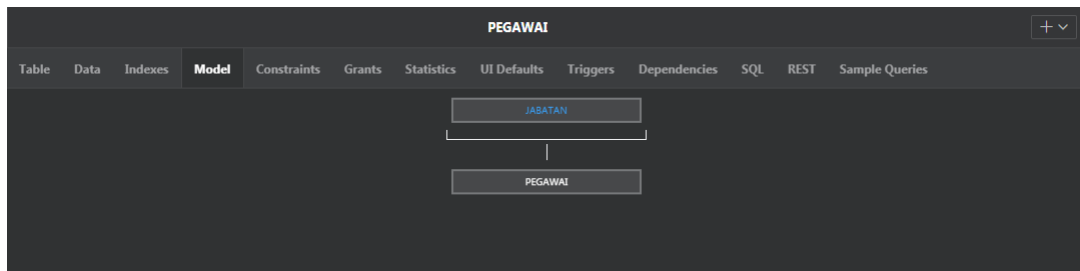
Relasi digunakan sebagai penghubung antar table yang saling berhubungan. Untuk Menghubungkan table menggunakan foreign key, foreign key di dapatkan dari primary key table.

Ketikkan query berikut di SQL Commands :

```
ALTER TABLE PEGAWAI ADD FOREIGN KEY (JABATAN_ID)  
REFERENCES JABATAN (JABATAN_ID)
```



Hasilnya akan seperti ini :



1.2.4 Trigger

Trigger Database adalah sesuatu perintah yang kita siapan untuk berjalan otomatis ketika perintah tertentu(insert, update, delete) dijalankan.

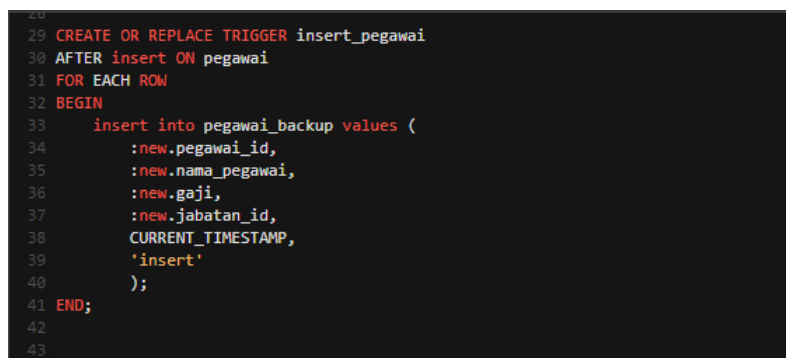
Membuat Trigger insert

1. Trigger insert pegawai

Trigger insert berfungsi jika ada memasukkan record baru ke dalam table pegawai maka recordnya juga masuk ke dalam table pegawai_backup

Ketikkan query berikut pada SQL Commands :

```
create or replace TRIGGER insert_pegawai
AFTER insert ON pegawai
FOR EACH ROW
BEGIN
insert into pegawai_backup values (
:new.pegawai_id,
:new.nama_pegawai,
:new.gaji,
:new.jabatan_id,
CURRENT_TIMESTAMP,
'insert'
);
END;
```



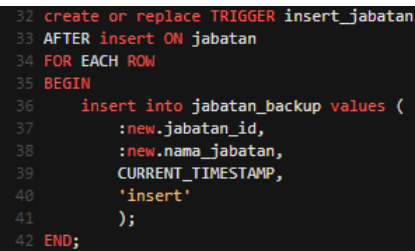
```
29 CREATE OR REPLACE TRIGGER insert_pegawai
30 AFTER insert ON pegawai
31 FOR EACH ROW
32 BEGIN
33     insert into pegawai_backup values (
34         :new.pegawai_id,
35         :new.nama_pegawai,
36         :new.gaji,
37         :new.jabatan_id,
38         CURRENT_TIMESTAMP,
39         'insert'
40     );
41 END;
42
43
```

2. Trigger insert jabatan

Trigger insert berfungsi jika ada memasukkan record baru ke dalam table jabatan maka recordnya juga masuk ke dalam table jabatan_backup

Ketikkan query berikut pada SQL Commands :

```
create or replace TRIGGER insert_jabatan
AFTER insert ON jabatan
FOR EACH ROW
BEGIN
insert into jabatan_backup values (
:new.jabatan_id,
:new.nama_jabatan,
CURRENT_TIMESTAMP,
'insert'
);
END;
```

A screenshot of a SQL code editor with a dark background. The code is written in a light-colored font and is numbered from 32 to 42. It defines a trigger named 'insert_jabatan' that fires after an insert operation on the 'jabatan' table. The trigger body inserts a record into the 'jabatan_backup' table with the new job ID, name, current timestamp, and the word 'insert'.

```
32 create or replace TRIGGER insert_jabatan
33 AFTER insert ON jabatan
34 FOR EACH ROW
35 BEGIN
36     insert into jabatan_backup values (
37         :new.jabatan_id,
38         :new.nama_jabatan,
39         CURRENT_TIMESTAMP,
40         'insert'
41     );
42 END;
```

Membuat Trigger update

1. Trigger update pegawai

Trigger update berfungsi jika record pada table pegawai diubah maka data sebelum di ubah akan masuk ke table pegawai_backup

Ketikkan query berikut pada SQL Commands :

```
create or replace TRIGGER update_pegawai
AFTER update ON pegawai
FOR EACH ROW
BEGIN
insert into pegawai_backup values (
:old.pegawai_id,
:old.nama_pegawai,
:old.gaji,
:old.jabatan_id,
CURRENT_TIMESTAMP,
```

```
'update'  
);  
END;
```

```
15 CREATE OR REPLACE TRIGGER update_pegawai  
16 AFTER update ON pegawai  
17 FOR EACH ROW  
18 BEGIN  
19     insert into pegawai_backup values (  
20         :old.pegawai_id,  
21         :old.nama_pegawai,  
22         :old.gaji,  
23         :old.jabatan_id,  
24         CURRENT_TIMESTAMP,  
25         'update'  
26     );  
27 END;  
28
```

2. Trigger update jabatan

Trigger update berfungsi jika record pada table jabatan diubah maka data sebelum di ubah akan masuk ke table jabatan_backup

Ketikkan query berikut pada SQL Commands :

```
create or replace TRIGGER update_jabatan  
AFTER update ON jabatan  
FOR EACH ROW  
BEGIN  
insert into jabatan_backup values (  
:old.jabatan_id,  
:old.nama_jabatan,  
CURRENT_TIMESTAMP,  
'update'  
);  
END;
```

```
20 create or replace TRIGGER update_jabatan  
21 AFTER update ON jabatan  
22 FOR EACH ROW  
23 BEGIN  
24     insert into jabatan_backup values (  
25         :old.jabatan_id,  
26         :old.nama_jabatan,  
27         CURRENT_TIMESTAMP,  
28         'update'  
29     );  
30 END;
```

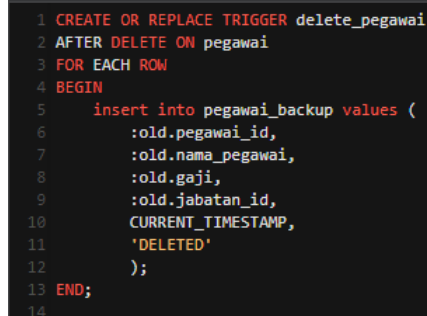
Membuat Trigger delete

1. Trigger delete pegawai

Trigger delete berfungsi jika record pada table pegawai dihapus maka akan recordnya akan masuk ke table pegawai_backup

Ketikkan query berikut pada SQL Commands :

```
create or replace TRIGGER delete_pegawai  
AFTER DELETE ON pegawai  
FOR EACH ROW  
BEGIN  
insert into pegawai_backup values (  
:old.pegawai_id,  
:old.nama_pegawai,  
:old.gaji,  
:old.jabatan_id,  
CURRENT_TIMESTAMP,  
'DELETED'  
);  
END;
```



```
1 CREATE OR REPLACE TRIGGER delete_pegawai  
2 AFTER DELETE ON pegawai  
3 FOR EACH ROW  
4 BEGIN  
5     insert into pegawai_backup values (  
6         :old.pegawai_id,  
7         :old.nama_pegawai,  
8         :old.gaji,  
9         :old.jabatan_id,  
10        CURRENT_TIMESTAMP,  
11        'DELETED'  
12    );  
13 END;  
14
```

2. Trigger delete jabatan

Trigger delete berfungsi jika record pada table jabatan dihapus maka akan recordnya akan masuk ke table jabatan_backup

Ketikkan query berikut pada SQL Commands :

```
create or replace TRIGGER delete_jabatan  
AFTER DELETE ON jabatan  
FOR EACH ROW
```

```

BEGIN
insert into jabatan_backup values (
:old.jabatan_id,
:old.nama_jabatan,
CURRENT_TIMESTAMP,
'DELETED'
);
END;

```

```

create or replace TRIGGER delete_jabatan
AFTER DELETE ON jabatan
FOR EACH ROW
BEGIN
    insert into jabatan_backup values (
        :old.jabatan_id,
        :old.nama_jabatan,
        CURRENT_TIMESTAMP,
        'DELETED'
    );
END;

```

1.2.5 View

View adalah perintah query yang disimpan pada database dengan suatu nama tertentu, sehingga bisa digunakan setiap saat untuk melihat data tanpa menuliskan ulang query tersebut.

Membuat View pada table pegawai_backup

1. View Insert

Ketikkan query berikut pada SQL Commands :

```

CREATE VIEW pegawai_insert AS SELECT*FROM pegawai_backup
WHERE CHANGED_TYPE='insert';

```

```

1 CREATE VIEW pegawai_insert AS SELECT*FROM pegawai_backup WHERE CHANGED_TYPE='insert';

```

2. View Update

Ketikkan query berikut pada SQL Commands :

```
CREATE VIEW pegawai_update AS SELECT*FROM pegawai_backup  
WHERE CHANGED_TYPE='update';
```

```
2 CREATE VIEW pegawai_update AS SELECT*FROM pegawai_backup WHERE CHANGED_TYPE='update';
```

3. View Delete

Ketikkan query berikut pada SQL Commands :

```
CREATE VIEW pegawai_delete AS SELECT*FROM pegawai_backup  
WHERE CHANGED_TYPE='DELETED';
```

```
3 CREATE VIEW pegawai_delete AS SELECT*FROM pegawai_backup WHERE CHANGED_TYPE='DELETED';
```

Membuat View pada table jabatan_backup

1. View Insert

Ketikkan query berikut pada SQL Commands :

```
CREATE VIEW jabatan_insert AS SELECT*FROM jabatan_backup  
WHERE CHANGED_TYPE='insert';
```

```
CREATE VIEW jabatan_insert AS SELECT*FROM jabatan_backup WHERE CHANGED_TYPE='insert';
```

2. View Update

Ketikkan query berikut pada SQL Commands :

```
CREATE VIEW jabatan_update AS SELECT*FROM jabatan_backup  
WHERE CHANGED_TYPE='update';
```

```
CREATE VIEW jabatan_update AS SELECT*FROM jabatan_backup WHERE CHANGED_TYPE='update';
```

3. View Delete

Ketikkan query berikut pada SQL Commands :

```
CREATE VIEW jabatan_delete AS SELECT*FROM jabatan_backup  
WHERE CHANGED_TYPE='DELETED';
```

```
CREATE VIEW jabatan_delete AS SELECT*FROM jabatan_backup WHERE CHANGED_TYPE='DELETED';
```

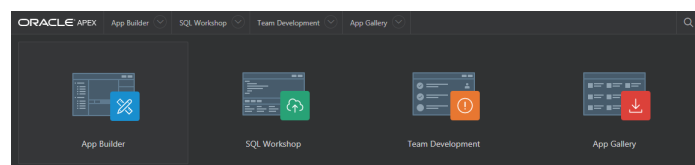
1.3 Membuat Aplikasi dari rancangan database

1.3.1 Daftar halaman yang akan dibuat

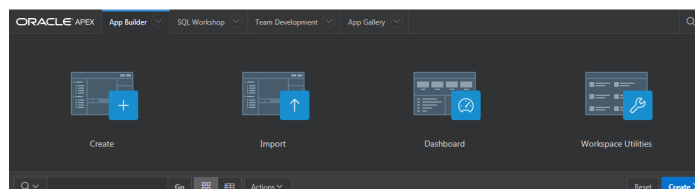
1. Membuat Halaman Form untuk menambahkan data pegawai
2. Membuat Halaman Form untuk menambahkan data jabatan
3. Membuat Halaman Interactive Report Untuk Menampung data penambahan jabatan(View insert table jabatan)
4. Membuat Halaman Interactive Report Untuk Menampung data pegawai masuk(View insert table pegawai)
5. Membuat Halaman Interactive Report Untuk Menampung data pegawai(table pegawai)
6. Membuat Halaman Interactive Report Untuk Menampung data jabatan(table jabatan)
7. Membuat Halaman Interactive Report Untuk Menampung data pegawai backup(table pegawai_backup)
8. Membuat Halaman Interactive Report Untuk Menampung data jabatan backup(table jabatan_backup)

1.3.2 Langkah-langkah membuat aplikasi

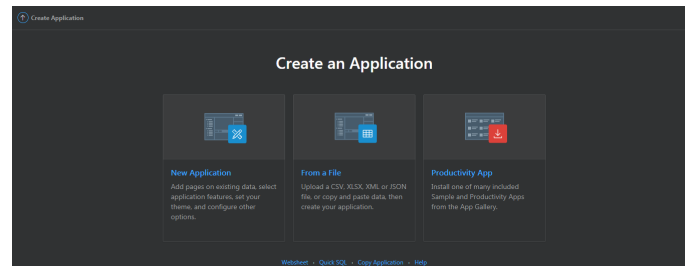
1. Klik App Builder



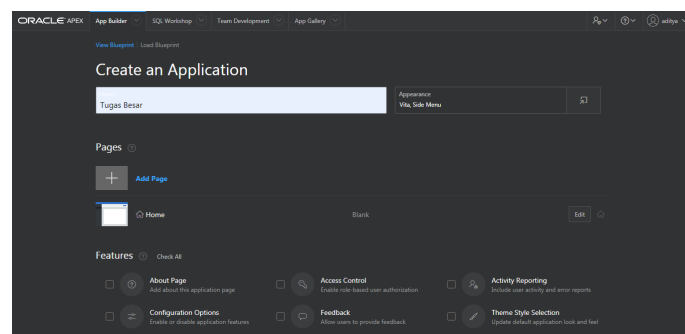
2. Lalu klik Create



3. Kemudian klik New Application

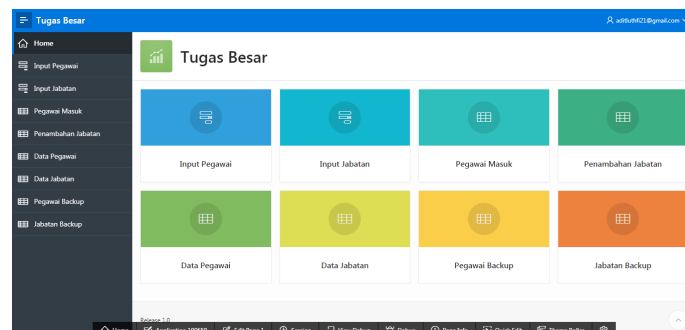


4. Setelah itu isi nama aplikasi yang ingin di buat



5. Selanjutnya klik Add Page untuk membuat halaman. Buatlah Halaman seperti yang sudah di jelaskan di atas.

6. Inilah tampilan Home aplikasi yang sudah dibuat dengan halaman-halaman seperti diatas.



Chapter 2

Akun

2.1 Login Apex

1. Workspace : RIBIB
2. Username : ADITLUTHFI21@GMAIL.COM
3. Password : ribib021

2.2 Login Aplikasi

1. Link Aplikasi :
<https://apex.oracle.com/pls/apex/f?p=100659:1:108731702524684::::>
2. Username : aditluthfi21@gmail.com
3. Password : ribib021